המחלקה למתמטיקה ולמדעי המחשב
פרויקט מסכם בהנדסת תעשיה וניהול

קביעת מועדי שחרור בפרויקט עם תזרים מזומנים ואילוצי משאבים באמצעות אלגוריתם PSO

חיבור זה מהווה חלק מהדרישות לקבלת תואר ראשון
בהנדסת תעשיה וניהול באוניברסיטה הפתוחה

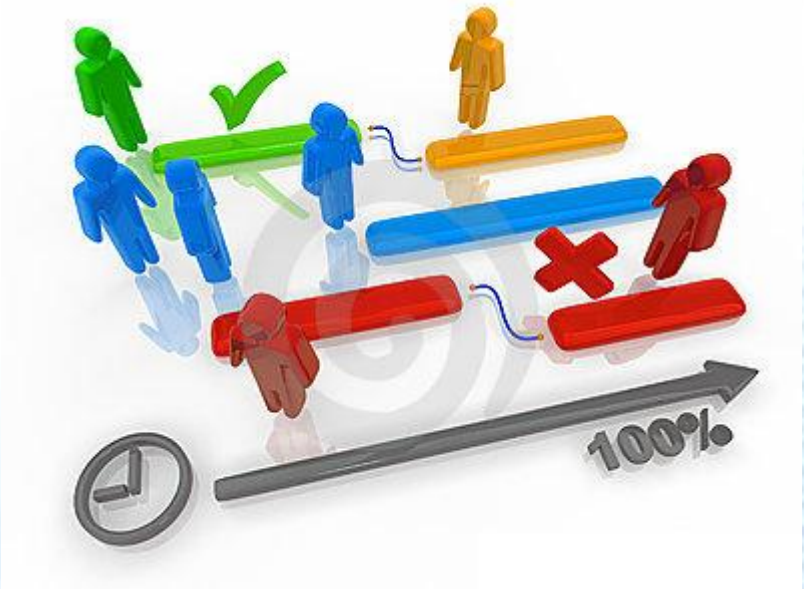
מאת
תום זמיר, ת.ז. 034596437
אורטל חובה, ת.ז. 038116786

אדר, התשע"ג **פברואר, 2013**

המחלקה למתמטיקה ומדעי המחשב

פרויקט מסכם בהנדסת תעשייה וניהול

קביעת מועדי שחרור בפרויקט
עם תזרים מזומנים ואילוצי משאבים
באמצעות אלגוריתם PSO



מאת:
תום זמיר 0345964437
אורטל חובה 038116786

אדר, התשע"ג פברואר, 2013

הצהרה

העבודה נעשתה בהנחייתו של ד"ר יובל כהן

וכן בהנחייתו של מר רן אתגר.

האוניברסיטה הפתוחה – המחלקה למתמטיקה ולמדעי המחשב.

החיבור מציג את עבודתנו האישית ומהווה

חלק מהדרישות לקבלת תואר ראשון בהנדסה.

כל טקסט ו/או תוצאה המבוססים על עבודות מחקר אחרות

מתועדים בציון המקור המדעי (Fully Referenced).

במסגרת הפרויקט, וכחלק בלתי נפרד ממנו,

יצרנו תוכנית מחשב ייחודית אשר נכתבה בשפת Java.

התוכנית נהגתה, אופיינה ונכתבה מהיסוד על ידינו ובמסגרת פרויקט זה בלבד.

תודות

אנו שמחים על כך שבפרויקט זה ניתנה לנו ההזדמנות לפתח את דמיוננו, חשיבתנו המדעית וכן את כושר ההמצאה שלנו. כמו כן, פרויקט זה היווה עבורנו הזדמנות לבחון במבט לאחור את הידע הנרחב והמגוון אותו רכשנו במהלך כל שנות לימודנו בתוכנית להנדסת תעשיה וניהול באוניברסיטה הפתוחה.
הערכתנו הרבה נתונה:

- **למנחה האקדמי הראשי, ד"ר יובל כהן:** אנו מודים לך מקרב הלב על תמיכתך לאורך כל הדרך, על ההכוונה וההנחיה וכמובן על העזרה בבחירת הרעיון לנושא הפרויקט! אנו שמחים שלקחת אותנו תחת חסותך והקדשת לנו מזמנך גם בזמנים עמוסים.
- **למנחה האקדמי, מר רן אתגר:** על העזרה והסיוע בכתיבה האקדמית. שימשת לנו זוג עיניים מנוסות נוסף ומקור לביקורת בונה אשר תרמה לליטוש ההצגה הנאותה של הפרויקט.
- **למרכז הקורס, מר רון וולף:** על עזרתך במציאת מנחה מתאים וכן על גילוי סבלנות, הבנה וגמישות מנהלית, מבלי לגרוע מרמתו האקדמית של הפרויקט.

תוכן עניינים

0. תקציר, עמ' 7.

1. מבוא, עמ' 8.

2. מטרת הפרויקט, עמ' 9.

3. סקר ספרות, עמ' 11.

3.1. תיאור בעיית התזמון של פרויקט מאולץ משאבית – RCPSP, עמ' 11.

3.1.1. הקדמה, עמ' 11.

3.1.2. ניסוחים שונים של בעיית RCPSP, עמ' 12.

3.1.3. ניסוח מתמטי של בעיית RCPSP עם תזרים מזומנים, עמ' 14.

3.2. אלגוריתמים אבולוציוניים, עמ' 18.

3.3. Particle Swarm Optimization – PSO, עמ' 20.

3.3.1. הקדמה, עמ' 20.

3.3.2. הגדרות וסימונים באלגוריתם PSO, עמ' 21.

3.3.3. אלגוריתם PSO הכללי, עמ' 21.

3.3.3.1. נוסחת עדכון המהירות – פירוט, עמ' 22.

3.3.4. Multigrouped PSO – MGPSO, עמ' 23.

3.4. מושגים בסיסיים בתוכנה, עמ' 26.

3.4.1. מבני נתונים, עמ' 26.

3.4.2. בו-זמניות, עמ' 29.

4. המודל המוצע לפתרון בעיית תזמון מועדי השחרור בפרויקט, עמ' 30.

4.1. תיאור של בעיית RDTP (Release Dates Timing Problem), עמ' 30.

4.1.1. הקדמה, עמ' 30.

4.1.2. בעיית RDTP הכללית, עמ' 30.

4.1.3. בעיית RDTP המצומצמת, עמ' 31.

4.1.4. ניסוח מתמטי של בעיית RDTP המצומצמת, עמ' 31.

4.2. פתרון בעיית RDTP הכללית באמצעות פתרון של בעיית RDTP המצומצמת, עמ' 33.

4.3. חיזוק כושר אלגוריתם PSO, עמ' 35.

4.4. התמודדות עם בעיית מרחב הפתרונות הבדיד, עמ' 37.

4.4.1. עטיפה לבעיית RCPSP, עמ' 38.

4.4.2. עטיפה לבעיית RDTP, עמ' 39.

4.4.3. שמירה על הכושר של האלגוריתם, עמ' 40.

4.5. לשם מה נחוץ לנו MGPSO, עמ' 40.

5. שיפור ביצועים, עמ' 44.
5.1. כללי, עמ' 44.
5.2. ההשפעה של MGPSO על יעילות האלגוריתם המוצע, עמ' 44.
5.3. שימוש במבני נתונים יעילים, עמ' 45.
5.3.1. שימוש ברשימה מקושרת, עמ' 47.
5.3.2. שימוש בטבלת גיבוב, עמ' 48.
5.3.3. פיתוח מבנה נתונים יעיל ייחודי, עמ' 48.
5.4. שימוש בבו-זמניות, עמ' 49.
6. מהלך המחקר, עמ' 51.
6.1. ניסוח מטרת המחקר, עמ' 51.
6.2. אופן המדידה והסקת המסקנות, עמ' 51.
6.2.1. אלגוריתמי הבקרה, עמ' 51.
6.2.2. מספר המדגמים ואופיים, עמ' 53.
6.2.3. קביעת NPVmax כבסיס לחישוב ציוני ה-POM, עמ' 54.
6.2.4. הסקת המסקנות, עמ' 54.
6.3. קבועים וסדרי גודל בשימוש, עמ' 55.
7. תוצאות המחקר, עמ' 56.
7.1. ערכי NPVmax, עמ' 56.
7.2. השוואה ביו ממוצע עוצמת הפתרונות לפי מדד POM, עמ' 56.
7.3. השוואה בין עקביות הפתרונות ע"פ סטיית התקן, עמ' 57.
7.4. מדידת שונות פנימית באלגוריתם ה-PSO, עמ' 58.
7.5. השוואה בין זמני הריצה הממוצעים (בשניות), עמ' 59.
8. מסקנות ודיון, עמ' 60.
9. סיכום, עמ' 65.
10. ביבליוגרפיה, עמ' 66.
11. נספחים, עמ' 69.
נספח א' – דוגמא לפלט התוכנית שפותחה במסגרת הפרויקט, עמ' 69.
נספח ב' – גרפים הכוללים את תוצאות האלגוריתם האקראי הפשוט, עמ' 71.
נספח ג' – טבלת תוצאות המקסימום שהושגו על פני כל ההרצות במחקר, עמ' 72.
נספח ד' – היסטוגרמות ציוני POM בחלוקה לאלגוריתמים וגודל הפרויקט, עמ' 74.
נספח ד' – דיסק מצורף המכיל: תוצאות הריצות של האלגוריתמים, עמ' 79.

0. תקציר

קביעת תזמון פעילויות בפרויקטים הינה בעיה מורכבת בפני עצמה. כאשר מביאים בחשבון פרויקט בעל מועדי שחרור תכולות, הבעיה שבקביעת מועדי שחרור אלו מוסיפה קושי חישובי לבעיית התזמון המקורית משום ששתי הבעיות מסובכות אחת בשנייה (Entangled). את התסבוכת (Entanglement) הזו אנו מכנים בשם RDTP – Release Dates Timing Problem. במסגרת פרויקט זה הצענו ומימשנו פתרון מטה-היוריסטי לבעיית RDTP באמצעות אלגוריתם המכונה PSO – Particle Swarm Optimization. הפתרון מומש כך שישאף להשיא את הענ"ן של פרויקט בעל תזרים מזומנים המתמודד עם אילוצי קדימויות ואילוצי משאבים. בחנו את כושרו של אלגוריתם ה-PSO אל מול אלגוריתמי בקרה פשוטים, סטוכסטיים ודטרמיניסטיים. אלגוריתם ה-PSO נמצא יעיל ביותר, אך הפתיעו בכושרם האלגוריתמים הפשוטים הדטרמיניסטיים. כוחם של האלגוריתמים הפשוטים הולך ונשחק ככל שפרויקט הוא בעל: גמישות במרווח בין מועדי השחרור וכן גמישות מבחינת אילוצי הקדימויות. לעומת זאת, כוחו של PSO נשאר גבוה עבור כל המקרים שבחנו.

1. מבוא

בעיות תזמון הינן מוקד התעניינות משמעותי בתחום של חקר ביצועים והנדסת תעשייה ומופיעות רבות בספרות עוד מתחילת שנות ה-50. המטרה העיקרית של פעולת התזמון הינה הקצאה יעילה של משאבים משותפים עבור פעילויות המתחרות עליהן, לאורך זמן. תשומת לב מיוחדת ניתנה לחקר של מציאת תזמון הולם לפעילויות פרויקטים בתחום התעשייה, הבניין, התוכנה וכו'. השיטה המוכרת ביותר לתזמון פעילויות, וגם הנאיווית ביותר, קרויה בשם "הנתיב הקריטי" – CPM – Critical Path Method והיא שמה לה למטרה למצוא את משך הפרויקט הקצר ביותר תוך עמידה באילוצי הקדימויות של הפעילויות. שיטה זו הינה קלה יחסית לפתרון ויעילה גם עבור בעיות בסדרי גודל גדולים מאוד אך היא מניחה אי-קיום אילוצי משאבים, מה שכמובן לא עולה בקנה אחד עם פרויקטים במציאות. מציאת תזמון אופטימאלי לפעילויות בפרויקט הנתון תחת אילוצי משאבים הינה בעיה הנקראת RCPSP – Resource Constrained Project Schedule Problem, והיא נמצאת בקטגוריה הקשה ביותר של בעיות קומבינטוריות אשר לא ניתן לפותרן בזמן פולינומיאלי (NP-hard). המשמעות היא שלא ניתן לחשב, בזמן סביר, פתרון אופטימאלי **מדויק** לבעיה מסדר גודל בינוני ומעלה (מעל כ- 30 פעילויות).

בתחילה, ניסו החוקרים לפתור את בעיית ה-RCPSP בשיטות מדויקות כגון "סעף וחסום". השיטות המדויקות הינן בעלות חשיבות רבה מבחינה תיאורטית ומחקרית ואכן ניתן לפתור בעזרתן בעיות בסיס, אך לעיתים הן גוזלות זמן רב מדי, אפילו לבעיות בסדר גודל צנוע. עם התקדמות הטכנולוגיה והמחקר בתחום חקר הביצועים, המיקוד עבר מהשיטות המדויקות אל עבר השיטות "המקורבות" או ה"היוריסטיות", בעיקר עבור פתרון בעיות מציאותיות גדולות אשר דרשו מענה בשטח בזמן קצר. שיטות חיפוש סטוכסטי לוקאלי כמו SA - simulated annealing נתנו מענה מקורב לבעיות אופטימיזציה קומבינטוריות. הן סיפקו גישה איתנה ליצירת פתרונות איכותיים לבעיות בסדר גודל מציאותי ותוך זמן סביר.

חלק משיטות החיפוש הסטוכסטי הלוקאלי נסמכות על עקרונות של בינה מלאכותית ואילו אחרות מבוססות על אנאלוגיות לתהליכים המתרחשים בטבע כמו תהליכים ביולוגיים או פיסיקליים. SA למשל, קיבלה את שמה משום שהיא מבוססת על תהליך החישול (annealing): חימום מוצק עד להתכתו ולאחר מכן קירורו ההדרגתי באופן כזה שמתמצק מחדש במבנה אחר, מסודר יותר. GA - Genetic Algorithm מבוסס על עקרונות האבולוציה והגנטיקה והמנגנון המייצר שיפורים מדור לדור. Ant Colony – שיטה המחקה את התנהגותם של הנמלים למציאת השבילים הטובים ביותר. שיטות אלו, לרוב פועלות בצורה של הפעלת מהלך ראשי בצורה איטרטיבית. בכל איטרציה, המהלך הראשי מכוון ומשפיע על פעולתן של היוריסטיקות פנימיות; על כן, השיטות הללו נקראות גם **מטה-היוריסטיות**. שיטות מטה-היוריסטיות באו לידי שימוש בפתרון קשת רחבה של בעיות (ביניהן גם קומבינטוריות) בהצלחה מרובה.

2. מטרת הפרויקט

מטרת העל:

בפרויקט מחקרי זה אנו מרחיבים את בעיית ה- RCPSP המוכרת ומוסיפים עליה את מימד קביעת מועדי השחרור (release dates). את הבעיה החדשה אנו מכנים בשם RDTP – Release Dates Timing Problem. גולת הכותרת של הפרויקט הינה כתיבה ומימוש תוכנית מחשב ייחודית אשר תהווה כלי יעיל לפתרון בעיות RCPSP ו-RDTP בעלות זיקה מרבית למציאות, מעבר לאילוצי הקדימויות;

- כמות פעילויות גבוהה.
- יותר מסוג משאב יחיד.
- אילוצי משאבים לכל/רוב הפעילויות.
- קיומן של פעילויות הדורשות השקעה כספית בתחילתן.
- קיומן של פעילויות המניבות תזרים מזומנים עם סיומן, על פני תקופה ספציפית.
- קבלת ערך רק במועד שחרור.
- התייחסות לריבית בחישוב הענ"ן של הפרויקט.

מוטיבציה:

מועדי שחרור הינם נחלתם של פרויקטי פיתוח ופרויקטים גמישים, המאפשרים ללקוח ולארגון המייצר לקבל ערך עוד בטרם סיום הפרויקט. מועדי השחרור מאפשרים גם היזון חוזר בין היצרן ללקוח, אשר מקבל לידיו תכולות ביניים בשלבים שונים במהלך חיי הפרויקט. מעבר למשמעות האסטרטגית שלהם, למועדי השחרור השפעה מכרעת על ערך הפרויקט ולכן השאלה היכן למקמם על ציר הזמן היא שאלה בלתי נמנעת. יחד עם זאת, טכניקות לקביעת מועדי שחרור למעשה אינן קיימות בספרות, יתכן משום הקושי החישובי הרב הכרוך בכך. מכיוון ששאלת תזמון מועדי הסיום רלוונטית עבור פרויקטים רבים ומכיוון שהיא למעשה חסרת מענה שיטתי מוצק – בחרנו במסגרת הפרויקט לנסח בצורה שיטתית את הבעיה, להציע לה דרך פתרון היוריסטית (ובכך למעשה לעקוף את מרב הקושי החישובי), וכן לממש את הפתרון ע"מ להדגים את יכולותיו.

שאלת המחקר:

נרצה לענות על השאלה האם ניתן לרתום את כוחו של אלגוריתם מטה-היוריסטי מסוג PSO על מנת לספק פתרונות איכותיים לבעיית קביעת מועדי השחרור. ובאופן ממוקד יותר: האם הכלי המטה-היוריסטי מבוסס ה-PSO אשר נפתח במסגרת פרויקט זה יניב תוצאות עדיפות ביחס לאלגוריתמים אחרים, פשוטים יותר, בפתרון בעיות תזמון מועדי שחרור.

הפרוצדורה:

ראשית, ננסח בצורה מתמטית את בעיות RCPSP ו-RDTP אותן אנו מתיימרים לפתור. לאחר מכן, נציע ונממש פתרון לבעיות באמצעות תוכנית מחשב. התוכנית תתבסס על אלגוריתם מטה-היוריסטי הנשען על שיטת "אופטימיזציית נחיל חלקיקים" – PSO - Particle Swarm Optimization. נר לרגלינו במהלך הפרויקט הינו האיזון שבין שמירה על פשטות הבעיה והפתרון לצורך ההצגה האקדמית שלהם, לבין התאמתם למציאות המורכבת. ע"מ לאמוד את תועלתו של אלגוריתם מבוסס ה-PSO, נערוך השוואה בינו לבין אלגוריתמים חמדניים מסוגים שונים על-פני מספר הרצות ועל-פני בעיות בגדלים שונים. הבעיות עליהן תיערך ההשוואה תהיינה מבוססות על בעיות בוחן סטנדרטיות מתוך ספריית PSPLIB, המקובלות בספרות האקדמית, ותיעשה ע"פ המדדים הבאים:

א. מידת קירוב הפתרון המתקבל לפתרון האופטימאלי (הידוע) לבעיה.

ב. עליונות הפתרון המתקבל בהשוואה לפתרון של אלגוריתם רנדומאלי טהור מצומצם (הרצה בודדת של פונקצית המטרה).

ג. עליונות הפתרון המתקבל בהשוואה לפתרון של אלגוריתם רנדומאלי טהור רחב (מס' הרצות של פונקצית המטרה כמספר ההרצות באלגוריתם ה-PSO).

ד. סיבוכיות האלגוריתם וזמן ריצה (CPU time).

הנחות המחקר:

במסגרת המחקר אנו מניחים מספר הנחות בנוגע לטבען של הפעילויות בפרויקט וכן לאופי תזרים המזומנים. ההנחות הללו מפורטות בפרקים הסוקרים את בעיות RCPSP ו-RDTP . בשלב זה נציין רק את ההנחות הבאות:

- המטרה המשותפת של שתי הבעיות הנ"ל הינה השאת ענ"ן הפרויקט.
- לאורך כל המחקר אנו מניחים כי לכל הפרויקטים ארבעה מועדי שחרור; שלושה מועדים במהלך חיי הפרויקט וכן מועד שחרור נוסף החופף למועד סיום הפרויקט.
- כמו כן, אנו מניחים כי קנס החריגה ממשך גרסה סטנדרטית זהה לכל הפרויקטים, ועומד על 100 ₪ ליום.

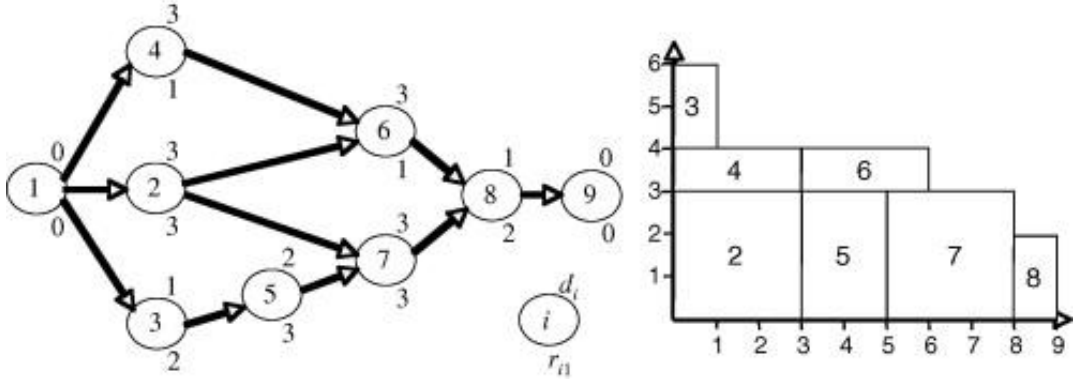
3. סקר ספרות

3.1 תיאור בעיית התזמון של פרויקט מאולץ משאבית - RCPSP

3.1.1 הקדמה

בעיית תזמון הפרויקט המאולץ משאבית (Resource Constrained Project Scheduling Problem) עוסקת בתזמון של פעילויות התלויות אחת בשנייה על פני רצף של זמן, כאשר לרשותן עומדים משאבים מוגבלים. תזמון פרויקטים חיוני במיוחד עבור חברות המייצרות ע"פ עקרון של "ייצור לפי הזמנה" (Make To Order), כלומר המוצרים אינם מיוצרים עד אשר מתקבלת הזמנה מהלקוח. גישה זו מאפיינת תעשיות כגון בניין, תוכנה ורכב, המייצרות מוצרים בעלי מידה גבוהה של התאמה ללקוח ספציפי. ברוב המקרים, על אנשי הפרויקט להתמודד עם משאבים מצומצמים הנובעים מגישה חסכונית של הנהלת הארגון ומרצון להגדיל את שולי הרווח. כמו כן עליהם לספק את המוצר הטוב ביותר עבור הלקוח ובזמן המהיר ביותר האפשרי, ע"מ להיות תחרותיים מול יצרנים אחרים.

בבעיה סטנדרטית מניחים כי הפרויקט מורכב מ-n,..,1 פעילויות, ולצורך הפשטות מניחים גם שתי פעילויות דמה; אחת כשורש הפרויקט (פעילות 0) והשנייה לניקוז הפעילויות (פעילות n+1). מקובל לייצג את מבנה הפרויקט באמצעות תרשים רשת AON (Activities On Nodes), כך שהצמתים מייצגים את הפעילויות ואילו הקשתות את אילוצי הקדימויות. כמו כן קיים סט התחלתי של משאבים מתחדשים, אשר עומד לרשות הפרויקט. לכל פעילות מציינים את משכה וכן את כמות המשאבים אשר היא דורשת. כל הנתונים (צריכה, זמנים וכו') נמדדים במספרים שלמים בלבד. המטרה המקובלת היא מציאת תזמון קצר ככל הניתן אשר עומד באילוצי הקדימויות כמו גם באילוצי המשאבים. בבעיה הבסיסית קיים רק משאב מתחדש יחיד וכן מניחים: **(א)**משך הפעילויות הינו דטרמיניסטי, **(ב)**לא ניתן לחלק פעילות לחלקים אלא יש לבצע אותה ברצף ו**(ג)**מידת צריכת המשאבים של פעילות הינה קבועה לכל אורך הפעילות.



איור 3.1.1.1 – תזמון פרויקט עם אילוץ משאבים. מתוך www.sciencedirect.com.

משמאל: רשת הפרויקט. מימין: תרשים המתאר את תזמון הפעילויות וכן את צריכת המשאבים שלהן.

בתוך כל צומת מופיע אינדקס הפעילות, מעליה משך הפעילות ומתחתיה הצריכה של הפעילות.

3.1.2. ניסוחים שונים של בעיית RCPSP

בספרות ניתן לפגוש את בעיית RCPSP בגרסאות רבות שונות ומגוונות; עם סימונים שונים והנחות שונות. בסעיף זה נסקור *בקצרה* מספר גישות שונות לבעיה.

- **RCPSP רב מצבי (MRCPSP: Multi-mode RCPSP):**

בעיית RCPSP הבסיסית מניחה כי פעילות מסוימת יכולה להתבצע באופן מסוים ויחיד מבחינת משכה וצריכת המשאבים שלה. בבעיית MRCPSP לפעילות יכולים להיות מספר אפשרויות בהן היא יכולה להתבצע . כל מצב מתאר שילוב אפשרי בין זמן ביצוע לצריכת משאבים המאפשרים לפעילות להתבצע.
הנחות MRCPSP יכולות להיות מנוסחות בצורה הבאה:
פעילות j חייבת להתבצע באחד מהאופנים האפשריים שלה, אשר מסומנים $1,\ldots,M_j$ כך ש-M_j הוא מספר המצבים האפשרי. משהתחילה פעילות להתבצע באחד מהאופנים שלה, עליה להמשיך ולהסתיים באותו המצב; שינויים בין מצבים בזמן ריצה של פעילות הם אסורים.
מקרה פרטי של Multi-mode RCPSP הוא Crashable mode ("מצבים הניתנים למחיצה"). בגרסה זו קיים קשר המרה בין צריכת המשאבים לבין משך הפעילות כך שניתן לקצר את משך הפעילות בתמורה לצריכה מוגברת של משאבים (לאו דווקא באופן ליניארי).

- **בעיית ההשקעה במשאבים (RIP: Resource investment problem):**

בבעיה זו, לפעילויות יש משך קבוע והן צורכות כמויות קבועות של סוגים שונים של משאבים מתחדשים בזמן ריצתן. אילוצי קדימויות קיימים בין הפעילויות השונות. בד"כ נתון מראש גם זמן הסיום המאוחר המותר של הפרויקט (deadline) וגובה הקנס לכל יום חריגה. המשאבים השונים הינם מתחדשים והם נרכשים בתחילת הפרויקט וכן לכל משאב קיימת עלות ליחידה. המטרה הינה למצוא את מועדי תחילתן של כל הפעילויות כך שיביאו להשלמת הפרויקט, תוך מזעור העלות הכוללת של המשאבים וקנס החריגה ממועד הסיום המאוחר. *למידע נוסף ראה מקור: [11].*

- **מרווחי זמן מכסימאליים/מינימאליים (Minimal and maximal time lags):**

בבעיית RCPSP בסיסית, פעילות יכולה להתחיל רק לאחר שכל קודמותיה הסתיימו. ניתן להרחיב מושג בסיסי זה של אילוץ קדימויות ע"י שימוש במושג "מרווח זמן מינימאלי" בין פעילות לפעילות העוקבת שלה. בשיטה זו, פעילות אינה יכולה להתחיל מיד לאחר שכל קודמותיה הסתיימו אלא עליה לחכות פרק זמן מינימאלי הקבוע מראש, למשל כצורך בזמן כינון (setup time). ניתן להרחיב אף יותר את אילוץ הקדימויות ולקבוע מרווח זמן מינימאלי, או מכסימאלי, בין תחילת/סיום פעילות לבין תחילת/סיום הפעילות הקודמת לה. ההרחבה של אילוץ הקדימויות מכונה Generalized precedence relations.
למידע נוסף ראה מקור: [14].

- **היקף משאבים תלוי זמן (Time-dependent resource capacities):**

בגרסתה השפוטה של RCPSP מניחים כי היקף כל משאב (יחידות בשימוש ושאינן בשימוש) הינו קבוע לכל אורך הפרויקט. הנחה זו יכולה להתברר כנוקשה מדי בחלק מן המקרים במציאות, כמו למשל תנודות במצבת העובדים עקב חופשים, תנודות במצבת מלגזות עקב טיפולים תקופתיים וכו'. קיימות שתי שיטות עיקריות לייצוג השינויים בהיקפי המשאבים: **(א)**הגישה הישירה - הגדרה של מספר היקפים שונים לכל משאב, כך שכל אחד מהם תקף לחלק מסוים בציר הזמן. **(ב)**הגישה העקיפה - הגדרת היקף יחיד לכל משאב (כרגיל) וכן הגדרה של פעילויות דמה עבור כל משאב בעלות צריכה מסוימת וזמני תחילה וסיום ידועים מראש. בצורה זו משיגים את האפקט של שינוי ההיקף בזמן אע"פ שהיקף המשאב כשלעצמו נשאר קבוע. *למידע נוסף ראה מקור: [11].*

- **תזרים מזומנים תלוי-זמן (time dependent cash flow):**

בבעיות RCPSP בעלות תזרים מזומנים מקובל להתייחס לתקבולים השונים כבלתי תלויים בזמן קבלתם. גישה זו אומנם פשוטה אך אינה מייצגת היטב את המציאות בה למועד קבלת הערך יש חשיבות. לגישה שבנדון יתכנו מספר פתרונות שונים ע"מ להפוך את התקבולים לתלויי-זמן:

- קביעת קנסות לכל יום המאוחר (מוקדם) מזמן יעד לסיום (תחילת) פעילות.
- קביעת תמריץ לכל יום המוקדם מזמן יעד לסיום פעילות.
- קביעת מקדם שחיקה לרווח המתקבל מפעילות, כפונקציה של הזמן בין קבלת הרווח לבין תחילת הפרויקט. יכול להיות לינארי או מעריכי.

למידע נוסף ראה מקורות: [12], [13], [16].

נציין שבפרויקט זה אנו עושים שימוש במקדם שחיקה מעריכי ע"י חישוב הערך הנוכחי הנקי (NPV) של הערך המתקבל בתום גרסה, כפי שיוצג בהמשך.

- **תזרים מזומנים בנקודות קבועות בזמן (Cash inflow at specific points in time):**

ברירת המחדל ב-RCPSP עם תזרים מזומנים היא שפעילויות נושאות הערך מניבות אותו מיד בסיומן, וכן גובה הערך או התקבול ידוע מראש. השיטה הנוכחית באה בניגוד מה למצב "הטבעי" שהערך מוצמד לפעילות המניבה אותו מבחינה כרונולוגית. בשיטה זו מצב העניינים הינו הפוך בכך שידוע מראש מועד התקבול אך הערך שלו אינו ידוע. הערך הינו פונקציה של אחוז המשך שהושלם מהפעילות המניבה אותו, באותה נקודת זמן. *למידע נוסף ראה מקור: [11].*

נציין שבעבודה זו אנו עושים שימוש ברעיון של נקודות קבועות בזמן לקבלת התקבולים, אולם איננו מאפשרים קבלת ערך עבור פעילות שאינה הושלמה לחלוטין.

- **מועדי שחרור וזמני יעד של פעילויות (Release of activities and deadlines):**

מועד שחרור של פעילות j (לא להתבלבל עם מועד שחרור גרסה) הוא המועד המוקדם ביותר בו יכולה פעילות j *להתחיל*. באופן דומה, תאריך יעד של פעילות j מציין את המועד שבו יכולה *להסתיים* פעילות j, לכל המאוחר. ניתן לפגוש בגישה זו בעיקר בבעיות RCPSP עם מעט או ללא אילוצי קדימויות. ניתן להתייחס לגישה זו כאל מקרה פרטי של RCPSP עם מרווחי זמן מינימאליים/מכסימאליים, אם מועדי ההתחלה/הסיום הינם קשיחים והרווחים נמדדים ביחס לזמנים של פעילויות הדמה: השורש והניקוז. אם מניחים שניתן לחרוג לכאן או לכאן תחת קנס הרי שניתן לראות בגישה זו מקרה פרטי של בעיית RCPSP עם תזרים מזומנים תלוי זמן. *למידע נוסף ראה מקור: [17].*

- **פיצול פעילויות (Preemptive schedule):**

בעיית RCPSP הבסיסית מניחה כי לא ניתן לפצל פעילויות, אלא יש לבצע אותן ברצף מרגע שהתחילו. גישות שונות מאפשרות פיצול של פעילויות תחת תנאים שונים, למשל:

- פיצול פעילות אפשרי בכל עת.
- פיצול פעילות אפשרי רק לאחר כל מספר שלם כלשהו של יחידות זמן.
- פיצול פעילות אפשרי רק x פעמים.
- פיצול אפשרי רק בנקודות קבועות על ציר הזמן.

ישנן גישות המגדילות לעשות ומאפשרות לא רק פיצול של פעילויות אלא אף ריצה במקביל של חלקים של אותה הפעילות. *למידע נוסף ראה מקור: [18].*

- **משאבים בלתי מתחדשים (Non-renewable resources):**

בבעיית RCPSP בסיסית קיים סוג אחד של משאב הנקרא משאב מתחדש, משום שהוא חוזר לקיבולת המקורית שלו כאשר הפעילויות המשתמשות בו מסתיימות. דוגמאות למשאבים מתחדשים הם עובדים ומכונות. בבעיות RCPSP שונות ניתן לפגוש לעיתים בסוג משאב נוסף – משאב בלתי מתחדש. דוגמא למשאב בלתי מתחדש הוא תקציב. נשים לב כי יש טעם להציג משאב כבלתי מתחדש באופן מפורש רק בהקשר של MRCPSP, אחרת אין לו תרומה לבעיה. *למידע נוסף ראה מקור: [15].*

3.1.3. ניסוח מתמטי של בעיית RCPSP עם תזרים מזומנים

בעיה טיפוסית של RCPSP היא הרחבה של בעיית הנתיב הקריטי כך שנוספת לה התייחסות לאילוצי משאבים הקיימים בפרויקט. כיוון שכך, אחת המטרות השכיחות בספרות הינה למצוא את תזמון הפעילויות המוביל ל**משך הזמן** הקצר ביותר של הפרויקט. אולם, בפועל המטרה האמיתית הינה למקסם את **הרווח** מהפרויקט, ואין זה מחויב המציאות ששני התזמונים יהיו זהים. בפרויקט זה, אנו מתייחסים אל בעיית RCPSP המורחבת, קרי יש לתת את הדעת לא רק לאילוצי המשאבים אלא גם לתזרימי המזומנים בפרויקט (כולל השקעות) וכמובן לשווי הכסף עצמו (הריבית). **המטרה הינה השאת הענ"ן של**

הפרויקט. אנו בחרנו להגדיר את הבעיה בדרך הבאה: ישנן A פעילויות בפרויקט ולכל פעילות יש זמן ביצוע מוגדר $(1 \le i \le A)$ d_i. פעילות אינה יכולה להתחיל טרם סיומן של הפעילויות שבהן היא תלויה. לרשות הפרויקט עומדים R סוגי משאבים (מתחדשים), ולכל אחד מהם רמה התחלתית $(1 \le j \le R)$ r_j. לכל פעילות ישנה רמה ידועה של צריכת משאבים $r_{i,j}$. לכל פעילות תיתכן* דרישה להשקעה כספית בגובה מסוים I_i ע"מ להתחילה וכן יכולה* להניב תזרים מזומנים בגובה מסוים c_i לתקופה מסוימת f_i לאחר סיומה.

**ערך הפרמטר יכול להיות אפס*.*

הנחות הנוגעות לפעילויות:

- משך הפעילויות הינו דטרמיניסטי.
- לא ניתן לחלק פעילות לחלקים, יש לבצע אותה ברצף.
- מידת צריכת המשאבים של פעילות הינה קבועה לכל אורך הפעילות.

הנחות הנוגעות לתזרים המזומנים:

- הריבית אינה משתנה במהלך חיי הפרויקט.
- הריבית הינה ליום.
- גובה התקבולים הנובע מפעילות מסוימת הינו קבוע במשך כל תקופת התזרים.
- השקעות בפעילויות הדורשות השקעה מתבצעות בנק' ההתחלה של הפעילות.
- הנחה חשובה נוספת הינה שכל זרם מזומנים מתקבל **מיד** בסיום הפעילות שמבטיחה אותו. הנחה זו תבוטל ותוחלף באחרת כאשר נרחיב את הבעיה כך שתכלול גם החלטה על מועדי השחרור בחיי הפרויקט.

נגדיר את הפרמטרים הבאים:

t – נקודת זמן. מניחים תחילת פרויקט בזמן t=0 וכן שהזמן הינו בדיד ונמדד בימים שלמים.

t_i – זמן תחילתה של פעילות i **זהו משתנה ההחלטה**.

t_e – זמן סיומה (ותחילתה) של הפעילות האחרונה. מניחים קיומה של פעילות end בעלת אורך 0.

$Duration = \{d_1, d_2, \ldots, d_i\}$ - וקטור משכי הפעילויות. $(d_i \ge 0)$

$\mathrm{Re}\,sources = \{r_1, r_2, \ldots, r_j\}$ - וקטור רמות המשאבים ההתחלתיות. $(r_j \ge 0)$

$Investment = \{I_1, I_2, \ldots, I_i\}$ - וקטור השקעה נדרשת בכל פעילות. $(I_i \ge 0)$

$Cash = \{c_1, c_2, \ldots, c_i\}$ - וקטור גובה התקבולים הצפוי בעת סיום כל פעילות. $(c_i \geq 0)$

$Flow = \{f_1, f_2, \ldots, f_i\}$ - וקטור משך זרימת התקבולים הנובע מכל פעילות. $(f_i \geq 0)$

$precidence = \begin{bmatrix} p_{1,1} & \cdots & p_{1,i} \\ \vdots & \vdots & \vdots \\ p_{i,1} & \cdots & p_{i,i} \end{bmatrix}$ – מטריצת הקדימויות.

$p_{i,w} = 1$ אם פעילות w קודמת לפעילות i. אחרת, $p_{i,w} = 0$.

$consumption = \begin{bmatrix} r_{1,1} & \cdots & r_{1,j} \\ \vdots & \vdots & \vdots \\ r_{i,1} & \cdots & r_{i,j} \end{bmatrix}$ – מטריצת צריכת המשאבים. $(r_{i,j} \geq 0)$

$run_{i,t}$ - ריצת פעילות בזמן נתון (פרמטר מחושב). $run_{i,t} = 1 \Leftrightarrow (t_i \leq t) \wedge (t_i + d_i \geq t)$.

אחרת $run_{i,t} = 0$.

α - גובה הריבית.

FV_i - ערך עתידי נקי של פעילות i , לזמן סיומה. $FV_i = -I_i \cdot (1+\alpha)^{d_i} + c_i \cdot \frac{1-(1+\alpha)^{-f_i}}{\alpha}$

NPV_i - ערך נוכחי נקי של פעילות i מהוון לזמן t=0. $NPV_i = \frac{FV_i}{(1+\alpha)^{t_i+d_i}}$

נתייחס למשוואת חישוב הענ"ן:

הענ"ן – ערך נוכחי נקי, הינו שיטת הערכה של שווי זרם מזומנים המתקבלים בנקודות זמן שונות. בבסיס השיטה עומדת ההנחה כי אילו כסף שיקבל הארגון בעתיד היה עומד לרשותו כבר היום, הוא היה משקיע אותו בשיעור התשואה הנתון (המסומן כאן ב- α) וכך מרוויח עליו את הריבית. לחישוב הערך הנוכחי של סכום עתידי קוראים תהליך היוון. המשוואה הבסיסית להיוון ערך עתידי חד פעמי שיתקבל בעוד t תקופות מהיום הינה: $PV = \frac{FV}{(1+\alpha)^t}$. באופן דומה אך הפוך, ניתן לקבוע שווי של ערך עתידי על בסיס ערך קיים: $FV = PV \cdot (1+\alpha)^t$. ערך נוכחי הופך לערך נוכחי *נקי* לאחר שמפחיתים ממנו את ההשקעה הנדרשת לשם קבלת התקבול: $NPV = -I + \frac{FV}{(1+\alpha)^t}$. כאשר מעוניינים לחשב ערך נוכחי של זרם

תקבולים השווים בערכם ל-c ומתקבלים על פני t תקופות, יש להשתמש במקדם ערך נוכחי סדרתי (נקרא גם מענ"ס): $A_t = c\frac{1-\frac{1}{(1+\alpha)^t}}{\alpha}$.

הערה: נשים לב שהמענ"ס מניח קבלת תקבול ראשון ב-t=1 ולא t=0. המשמעות בהקשר של סיום פעילות היא שהתקבול יתקבל תקופה אחת לאחר סיום הפעילות. *ניתן* לתקן את חישוב ה-NPV של פעילות כך שהתקבול הראשון מתקבל מיד בזמן סיומה בצורה הבאה:

$$FV_i = -I_i \cdot (1+\alpha)^{d_i} + c_i \cdot \frac{1-(1+\alpha)^{-f_i}}{\alpha} + c_i - \frac{c_i}{(1+\alpha)^{f_i}}$$

התיקון מתבצע ע"י הפחתה של התקבול בתקופה f_i (התקבול האחרון) ובמקומו הוספה של תקבול המתקבל בזמן t=0.

לשם הפשטות, לא נקטנו בתיקון זה משום שהוא אינו עקרוני לצורך הצגה ומימוש של המודל המוצע. כמו כן בלאו הכי ניתן להניח עיכוב של תקופה אחת בין סיום הפעילות לבין מסירתה ללקוח וקבלת הערך.

כעת ננסח את פונקציית המטרה:

(1) $Maximize \quad \sum_{i=1}^{A} NPV_i$

$Subject \quad to:$

(2) $\forall i \forall w: p_{i,w} \cdot (t_w + d_w) \le t_i \qquad (1 \le i, w \le A)$

(3) $\forall t \forall j: \sum_{i=1}^{A} run_{i,t} \cdot c_{i,j} \le r_j \qquad (0 \le t \le t_e, 1 \le j \le R)$

ביטוי (1) הינו סכום הענ"נים של כל הפעילויות בפרויקט ולפיכך מבטא גם את הענ"ן של הפרויקט עצמו.

ביטוי (2) הינו האילוץ השומר על קדימויות הפעילויות. עבור כל פעילות w הקודמת לפעילות i, חייבת להסתיים טרם תחילתה של פעילות i.

ביטוי (3) הינו האילוץ השומר על צריכת משאבים כך שלא תחרוג מהרמה הנתונה. עבור כל יום ממשך הפרויקט, ועבור כל סוג משאב בנפרד, סך צריכת המשאבים של כל הפעילויות הרצות באותו הזמן לא יעלה על הרמה המותרת.

3.2. אלגוריתמים אבולוציוניים (EA - Evolutionary Algorithms)

- **כללי:**

אלגוריתם אבולוציוני הינו אלגוריתם מטה-היוריסטי מבוסס אוכלוסיה. ה"אוכלוסיה" מורכבת מפתרונות אפשריים לבעיה שבנדון והם משולים ל"פרטים". אלגוריתם אבולוציוני משתמש במנגנונים לשינוי התכונות של הפרטים באוכלוסיה בתהליך הדרגתי (או אבולוציוני) הנועד לקרב את הפרטים לפתרון האופטימאלי של הבעיה. אלגוריתמים אבולוציוניים שונים שואבים השראה מתהליכים פיסיקליים או ביולוגיים שונים כמו למשל: הִתרבות, מוטציה, ברירה טבעית, דפוסי התנהגות של בעלי חיים וכו'. נעיר כי במקורות שונים בספרות נעשית הבחנה בין אלגוריתם מבוסס אוכלוסיה לבין אלגוריתם אבולוציוני. ההבחנה נעשית על סמך השאלה האם פרטים בעלי כושר נמוך נדחים או מסולקים מהאוכלוסייה. אם כן, אזי האלגוריתם אבולוציוני, אחרת הינו מבוסס אוכלוסיה. נשים לב שלפי גישה זו האלגוריתם האבולוציוני הוא מקרה פרטי של אלגוריתם מבוסס אוכלוסיה. אנו לא נשתמש בהבחנה זו היות וברוב המכריע של המקורות מסווג אלגוריתם ה- PSO כאלגוריתם אבולוציוני, אע"פ שאינו דוחה פרטים בעלי כושר נמוך.

אלגוריתמים אבולוציוניים מצטיינים בד"כ במציאת פתרונות מקורבים לפתרון אופטימאלי למגוון רחב של בעיות בעיקר מכיוון שהם אינם מניחים הנחות מוקדמות באשר לבעיה עצמה או למרחב הפתרונות שלה. הצלחתם של האלגוריתמים האבולוציוניים באה לידי ביטוי בתחומים רבים כמו: הנדסה, ביולוגיה, כלכלה, שיווק, חקר ביצועים, פיסיקה ועוד.

- **הגדרה מתמטית:**

אלגוריתם אבולוציוני בסיסי ניתן להגדרה מתמטית בצורה הבאה:

$$P' = m(f(P))$$

כך ש:

P - הוא סט הפרטים המרכיב את האוכלוסייה בתחילת איטרציה.

P' – סט הפרטים המרכיב את האוכלוסייה בסוף האיטרציה.

f – פונקצית המטרה. נקראת גם fitness function כיוון שהיא משולה למדידת כושרו או טיב מיקומו של הפרט.

m – פונקציה המפעילה מניפולציה כלשהי על האוכלוסייה ומשנה אותה, בהתאם לערך המתקבל מ-f.

מקובל כי מספר האיטרציות הינו נתון לבחירת המיישם וכן שהפרטים באוכלוסיה הראשונית נוצרים באופן אקראי אחיד.

בהינתן סט פרטים באיטרציה כלשהי X, מקובל לכנות את סט הפרטים של האיטרציה ה-(X-1) בשם "הורים" או "אבות". אלגוריתם אבולוציוני עושה שימוש במידע האגור אצל "ההורים" ומנחיל אותם הלאה לפרטים הבנים. המידע הנ"ל יכול להיות מופק באופן מפורש או באופן בלתי מפורש, באמצעות

הפונקציה m. באמצעות מידע זה יכול האלגוריתם האבולוציוני להגיע לפתרונות שאינם מושגים באמצעות טכניקות שונות אחרות.

- **דוגמא למודל נפוץ: אבולוציה טבעית:**

כאמור, אלגוריתמים אבולוציוניים שונים ממשים בדרכים שונות את המנגנון m. כעת נדגים שימוש בעקרונות המודל הבסיסי לתיאור מודל גנרי מקובל השואב השראה מתהליך האבולוציה שקורה בטבע. במודל זה מיושמים שני מנגנונים המשפיעים על האוכלוסייה בטבע: ברירה טבעית ומוטציה.

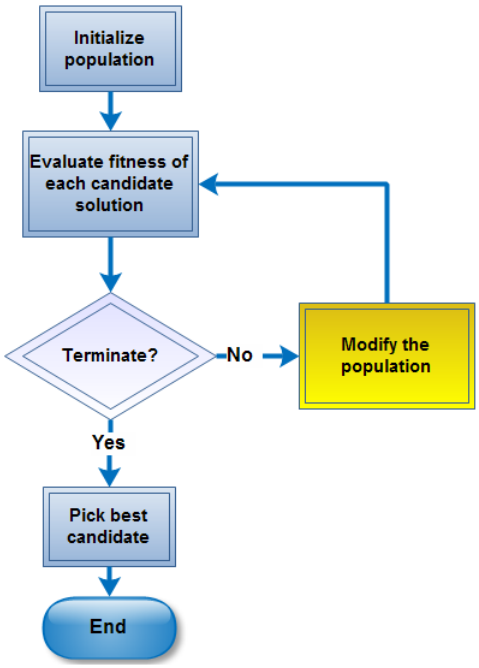
$$P' = \mu(s(f(P)))$$

μ - פונקצית מוטציה (mutation): יוצרת שינויים רנדומאליים בחלק מן הפרטים באוכלוסיה.

s – פונקצית ברירה (selection): הסרת פרטים בעלי כושר נמוך והחלפתם בעותקים של פרטים בעלי כושר גבוה מאיטרציה קודמת.

f, p, p' – כמוזכר במודל הבסיסי.

הטכניקות המסוימות בהן פועלות פונקצית המוטציה והברירה יכולה ללבוש צורות שונות, עם יתרונות וחסרונות ספציפיים לכל שיטה. ניתן לזהות אלמנט נוסף חשוב הקיים בטבע אך לא מופיע במודל זה – שילוב תכונות של הורים ליצירת פרט חדש טוב יותר. זהו אלמנט אשר מורכב יותר למימוש אך הוא נפוץ מאוד. אלגוריתמים המממשים אותו נקראים אלגוריתמים גנטיים (GA – Genetic Algorithms) והם שואבים השראה מתהליך הרבייה המתרחש בטבע באמצעות הגנים. אלגוריתמים גנטיים הם תת-קבוצה של אלגוריתמים אבולוציוניים. תיאור מדויק יותר של היישום של הפונקציות μ ו-s, וכן מימוש מנגנון הגנטיקה חורגים מההיקף המוגדר לפרויקט זה. הקורא המתעניין מופנה לעיין במקורות: [20], [21].



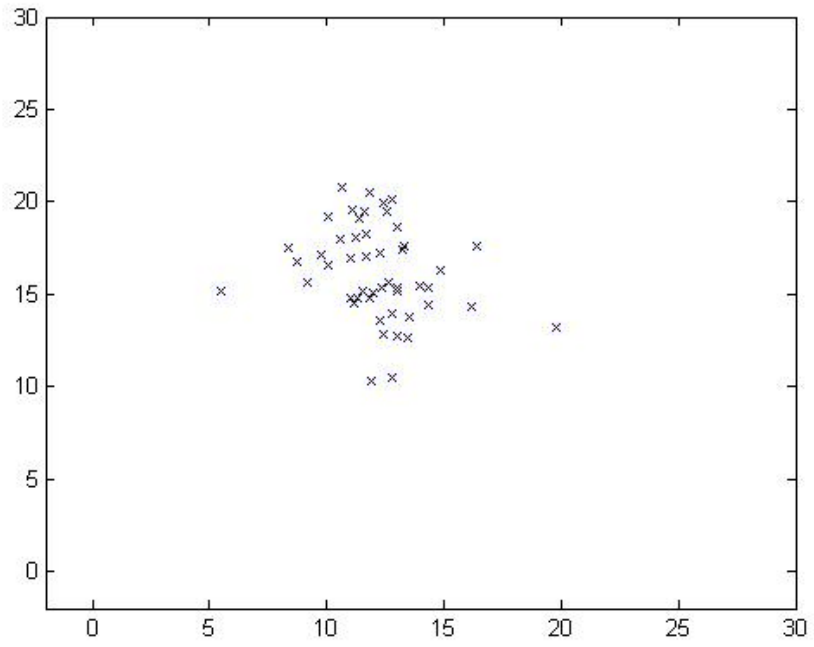
איור 3.2.1 – סכמה של אלגוריתם אבולוציוני/מבוסס אוכלוסיה. שלב המניפולציה על האוכ' ייחודי לכל שיטה.

3.3 Particle Swarm Optimization – PSO

3.3.1 הקדמה

PSO, אופטימיזציית "נחיל חלקיקים", הינה טכניקה היוריסטית מאוחרת יחסית לפתרון בעיות אופטימיזציה. הטכניקה פותחה לראשונה בשנת 1995 ע"י ד"ר אברהרט (Eberhart) וד"ר קנדי (Kennedy) . אלגוריתם PSO נחל הצלחה גדולה מאוד בפתרון בעיות אופטימיזציה בתחומים מגוונים הודות לקלות המימוש שלו ויעילותו החישובית. למרות זאת, שימוש באלגוריתם PSO לפתרון בעיות אופטימיזציה קומבינטוריות הוא פחות שכיח. אחת הסיבות לכך היא שבעיות מסוג זה הינן בעיות בעלות מרחב פתרונות בדיד וסופי (discrete) ואילו אלגוריתם PSO תוכנן במקור להתמודד עם בעיות בעלות מרחב פתרונות רציף (continuous) ואינסופי.

הרעיון שביסודו של PSO מבוסס על תצפיות על ההתנהגות החברתית של בעלי חיים כגון ציפורים ודגים החיים בלהקות, וכן על תיאוריות של התנהגות המונים. שמה של השיטה ניתן לה כיוון שהיא מדמה את הפתרונות האפשריים ההתחלתיים לפרטים, או חלקיקים, אשר להם מהירות ומיקום מסוימים. החלקיקים נעים במרחב הפתרונות של הבעיה, הנקרא גם "איזור החיפוש". התנועה של כל חלקיק מושפעת מהמיקום הטוב ביותר הקרוב אליו וגם ע"י מיקומים טובים אחרים במרחב הפתרונות אשר נמצאו ע"י חלקיקים אחרים. הציפייה היא שלאחר מספר איטרציות, החלקיקים ינועו כנחיל אל עבר הפתרונות הטובים ביותר.



איור 3.3.1.1 – מבט על של התכנסות חלקיקים במרחב פתרונות דו-מימדי 30X30. מתוך www.softpedia.com.

3.3.2. הגדרות וסימונים באלגוריתם PSO

- האלגוריתם הינו מטה-היוריסטי: מניח מעט מאוד הנחות על הבעיה, או בכלל לא, ומסוגל לחפש פתרונות במרחב גדול מאוד של מועמדים.
- פונקצית המטרה היא מהצורה: $\min f : R^n \to R$.

הפונקציה מקבלת וקטור של מספרים ממשיים המהווה פתרון אפשרי. הפונקציה מייצאת כפלט מספר ממשי (יחיד) המהווה את הערך של הפונקציה עבור אותו פתרון.

- מטרת האלגוריתם היא מציאת פתרון a עבורו $f(a) \le f(b)$ לכל b באזור החיפוש, כלומר מציאת מינימום גלובלי. עבור מציאת מקסימום גלובלי יש להפעיל האלגוריתם על $f' = -f$.
- הגדרת משתנים:
  - S – מספר החלקיקים בנחיל.
  - $x_i \in R^n$ - מיקומו של חלקיק i באזור החיפוש.
  - $v_i \in R^n$ - "מהירות" חלקיק i .
  - $p_i \in R^n$ - המיקום הידוע הטוב ביותר שחלקיק i היה בו.
  - $g \in R^n$ - המיקום הידוע הטוב ביותר שחלקיק כלשהו היה בו.

3.3.3. אלגוריתם PSO הכללי

בלוק אתחול מיקום ומהירות:

1. **עבור** כל חלקיק i = 1…S **בצע**:

1.1. אתחל את מיקום החלקיק עם וקטור רנדומאלי המתפלג באופן אחיד: $x_i \sim U(b_{lo}, b_{up})$, כך ש b_{lo}, b_{up} הם הגבולות (**b**oundaries) העליון והתחתון של מרחב הפתרונות.

1.2. אתחל את המיקום הטוב ביותר של החלקיק להיות מיקומו ההתחלתי: $p_i \leftarrow x_i$.

1.3. אם $f(p_i) < f(g)$ אזי עדכן את המיקום הטוב ביותר הכללי: $g \leftarrow p_i$.

1.4. אתחל את מהירות החלקיק: $v_i \sim U(-|b_{up} - b_{lo}|, |b_{up} - b_{lo}|)$

בלוק תנועה:

2. **בצע** כל עוד קריטריון עצירה אינו מתמלא (מס' איטרציות, פתרון עם תוצאה נאותה וכו'):

2.1. **עבור** כל חלקיק i = 1…S **בצע**:

2.1.1. **עבור** כל מימד d = 1…n **בצע**: *d is for dimension*

2.1.1.1. הגרל מקדמים $r_p, r_g \sim U(0,1)$ *p is for particle. g is for global*

2.1.1.2. עדכן את מהירות החלקיק: *זה החלק החשוב. ראה הסבר מטה*

$$v_{i,d} \leftarrow \omega \cdot v_{i,d} + \varphi_p \cdot r_p \cdot \left(p_{i,d} - x_{i,d}\right) + \varphi_g \cdot r_g \cdot \left(g_d - x_{i,d}\right)$$

2.1.2. עדכן את מיקום החלקיק: $x_i \leftarrow x_i + v_i$.

2.1.3. **אם** $f(x_i) < f(p_i)$ **אזי**:

2.1.3.1. עדכן את המיקום הטוב ביותר של החלקיק: $p_i \leftarrow x_i$.

2.1.3.2. אם $f(p_i) < f(g)$ אזי עדכן את המיקום הכללי הטוב ביותר $g \leftarrow p_i$.

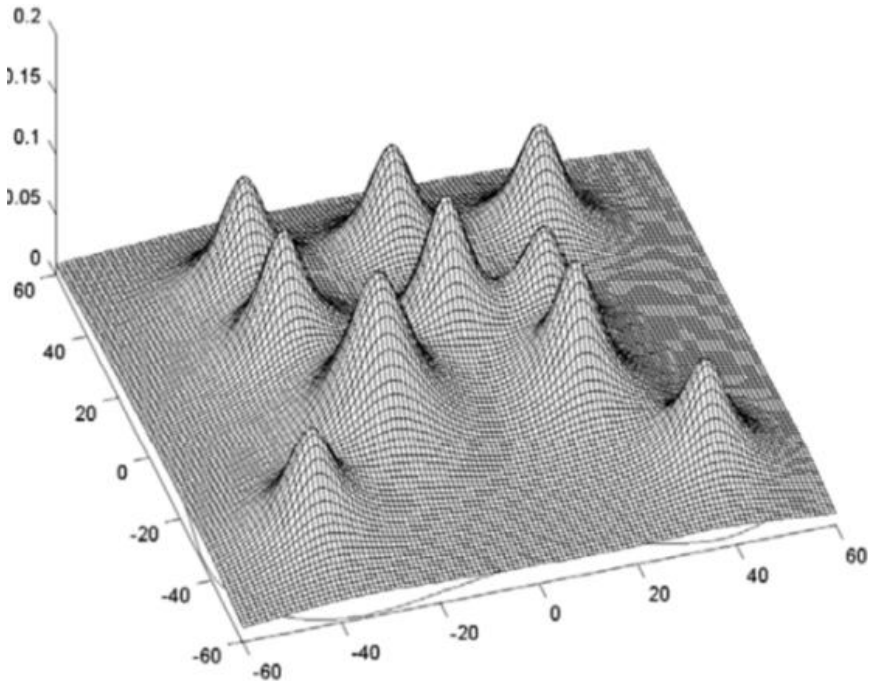
3. החזר את g .

3.3.3.1. **<u>נוסחת עדכון המהירות של החלקיק – פירוט:</u>**

- v_i הינה מהירות במובנה הווקטורי (velocity), כלומר יש לה גם גודל (speed) וגם כיוון.
- הפרמטרים $\omega, \varphi_p, \varphi_g$ נבחרים ע"י המשתמש ומשפיעים על התנהגות האלגוריתם ויעילותו.
- $\omega \cdot v_i$ - רכיב שימור המהירות הנוכחית. ω - המידה בה נרצה לשמר את המהירות הנוכחית של החלקיק, נקרא גם "מקדם האינרציה". מקדם האינרציה משמש לשליטה על מידת היותו של החלקיק "חקרן" (explore) ו/או "נצלן" (exploit). מקדם אינרציה גבוה גורם לחלקיקים לשמור על מהירות גבוהה ומונע מהם "להיתקע" באופטימום מקומי. לעומת זאת, מקדם אינרציה קטן גורם לחלקיקים לנוע לאט יותר מה שמאפשר להם לנצל ביתר יעילות את סביבת הפתרונות הקרובה אליהם.
- $\varphi_p \cdot r_p \cdot \left(p_i - x_i\right)$ - רכיב המהירות בכיוון המקום הטוב ביותר האחרון של החלקיק.
  - $\left(p_i - x_i\right)$ - וקטור לכיוון P (מרחק גדול יותר ← מהירות גבוהה יותר לאותו כיוון).
  - φ_p - המידה בה נרצה שמיקום P ישפיע על מהירות החלקיק.
  - r_p - רכיב רנדומאלי של השפעת P על מהירות החלקיק.
- $\varphi_g \cdot r_g \cdot \left(g - x_i\right)$ - רכיב מהירות בכיוון המיקום הכללי הטוב ביותר.
  - $\left(g - x_i\right)$ - וקטור לכיוון g (מרחק גדול יותר ← מהירות גבוהה יותר לאותו כיוון).
  - φ_g - המידה בה נרצה שמיקום g ישפיע על מהירות החלקיק.
  - r_g - רכיב רנדומאלי של השפעת g על מהירות החלקיק.

3.3.4. Multigrouped PSO – MGPSO

MGPSO היא טכניקה מתקדמת של שימוש באלגוריתם PSO המקורי אשר מציידת אותו ביכולת למצוא מספר פתרונות אופטימאליים בריצה אחת בודדת. MGPSO ישים במיוחד על פונקציות רבות-אופנים (multimodal functions) אשר להן מספר נקודות אופטימום מקומיות אשר קרובות או שוות בערכן לאופטימום הגלובלי של הבעיה.



איור 3.3.4.1 – המחשה של מרחב פתרונות של פונקציה רבת אופנים. מתוך [9].

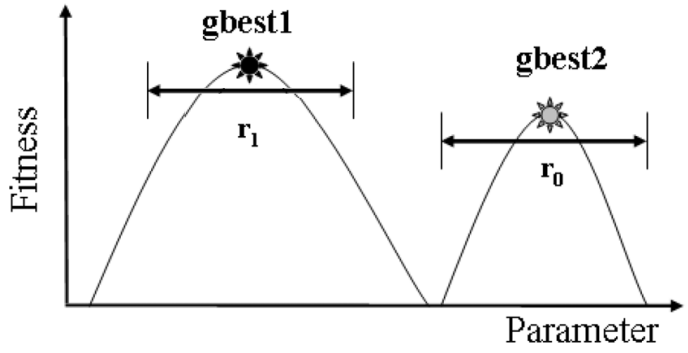
MGPSO נקרא כך כיוון שהוא מדמה מספר קבוצות של פרטים, או חלקיקים, המתחרים ביניהם על מיקומים טובים במרחב נתון. כל קבוצה פועלת על פי העקרונות של אלגוריתם PSO המקורי, אך לריצה של כמה קבוצות במקביל יש יתרונות שאינם קיימים בריצה טורית של PSO רגיל. היתרון המיידי של MGPSO הוא היכולת להחזיר מספר תוצאות בריצה אחת בצורה יעילה תוך הוספת מינימום עקרונות חדשים לאלו הקיימים ב-PSO המקורי. כעת נראה כיצד האלגוריתם משיג יתרון זה.

א. הגדרות בסיסיות: ראשית, MGPSO מניח שאין שינוי בכמות החלקיקים או מספר האיטרציות ממה שהוגדר במקור ע"י המיישם של אלגוריתם ה- PSO, כך שאין תוספת ניכרת של סיבוכיות או זמן ריצה. האלגוריתם דורש מהמיישם לקבוע מראש את מספר הקבוצות, m וכן את גודל הטריטוריה של כל קבוצה המבוטא כאחוז מאורכו של מרחב הפתרונות, t. לאחר מכן כלל החלקיקים נחלקים לקבוצות, כך שבכל קבוצה אותו מספר של חלקיקים. החלקיקים מקבלים מיקום רנדומאלי וכן מהירות התחלתית רנדומאלית.

ב. קביעת טריטוריות: לכל קבוצה נקבעת טריטוריה. הטריטוריה קבועה מבחינת גודלה אך היא

"נודדת" כאשר החלקיקים של הקבוצה מוצאים שטחים טובים יותר. לחלקיקים מותר "לפלוש" לטריטוריה של קבוצה אחרת אך אין הם רשאים להתיישב בה. השיטה בה נקבעת הטריטוריה לכל קבוצה היא כדלהלן:

- מחשבים עבור כל (מיקום של-) חלקיק את ערך פונקצית המטרה שלו.
- מרכיבים רשימה ממוינת של המיקומים, כך שהמיקום הראשון ברשימה הוא הטוב ביותר.
- מקצים את המיקום הראשון ברשימה לקבוצה הראשונה. מיקום זה הוא מרכז הטריטוריה של הקבוצה. הטריטוריה כוללת את כל הנקודות שמרחקן לא עולה על t מנקודת המרכז.
- בוחנים את המיקום הבא ברשימה; אם לא נמצא בתוך טריטוריה שכבר הוקצתה, מקצים אותו לקבוצה השנייה. אם נמצא בתוך הטריטוריה, בוחנים את הערך הבא וכך הלאה. בשיטה זו מקצים טריטוריות לכל שאר הקבוצות.

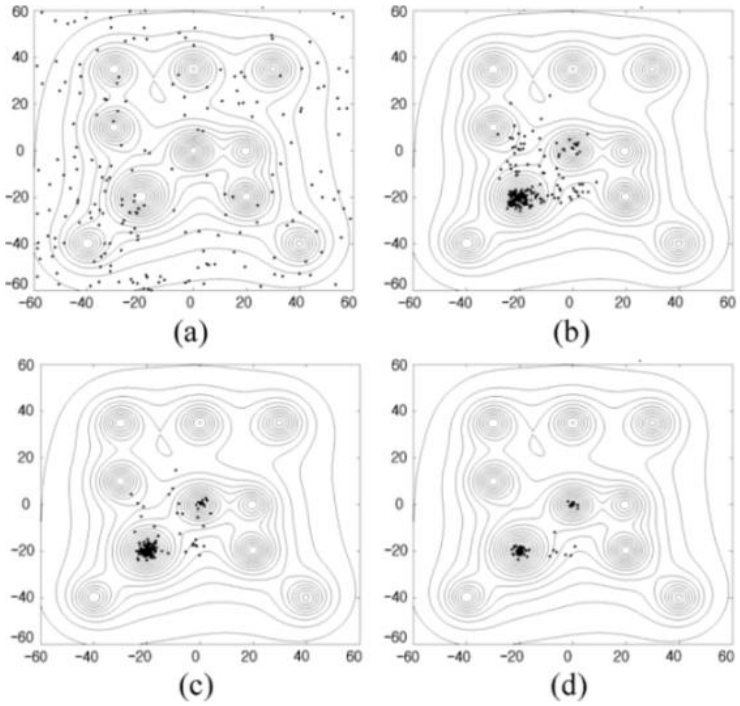


איור 3.3.4.1.1 – המחשה של טריטוריות של קבוצות ב-MGPSO. מתוך [10].

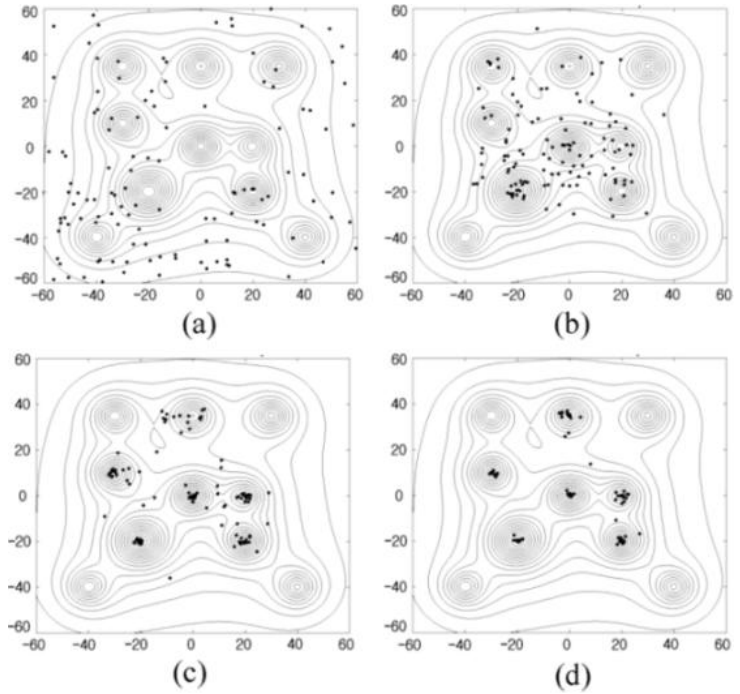
ג. **תנועה:** כמו ב-PSO, החלקיק נע במהירות בעלת 3 רכיבים: רכיב האינרציה, רכיב בכיוון הערך המיטבי האישי ורכיב בכיוון הערך המיטבי *הקבוצתי* במקום הערך המיטבי הגלובאלי. הבדל חשוב נוסף הוא שהחלקיק לא יכול לעדכן את הערך המיטבי האישי/הקבוצתי שלו במידה והוא נמצא באותה העת בטריטוריה של קבוצה אחרת. במקור [9] המחברים אף הגדילו לעשות והוסיפו רכיב רביעי לווקטור המהירות, רכיב "דחייה" אשר אמור לדחוף את החלקיק הרחק מטריטוריות זרות. למעשה, השגנו את האפקט המבוקש של MGPSO גם ללא מימוש של רכיב הדחייה.

היתרון של MGPSO נובע מהעובדה שהקבוצות השונות לא פועלות כל אחת בעולם משלה אלא הן מתקשרות אחת עם השנייה. בראשית הריצה, מרכזי הטריטוריות נקבעים על סמך מיקומיהן של כלל החלקיקים, על פני כלל הקבוצות. זהו המפתח הראשון לזיהוי של מספר נקודות אופטימום מקומי. המפתח השני הוא מושג הטריטוריה המאלץ כל קבוצה להתכנס לאזור שאינו חופף לאזורי כינוס של קבוצות אחרות. החיסרון שלו הוא הירידה בכושר של כל אחת מהקבוצות למצוא פתרונות טובים, ככל שיש יותר

קבוצות. החיסרון נובע מכך שעל החלקיקים להתפזר בין יותר קבוצות ולכן יש פחות מהם בכל קבוצה. ככל שישנם פחות חלקיקים בקבוצה כך פוחת הסיכוי שלהם לפגוש בפתרונות עדיפים.



איור 3.3.4.1.2 – המחשה של התכנסות חלקיקים ב-PSO רגיל בפונקציה מרובת אופנים. מתוך [9].



איור 3.3.4.1.3 – המחשה של התכנסות חלקיקים ב-MGPSO באותה הפונקציה. מתוך [9].

3.4. מושגים בסיסיים בתוכנה

כחלק מרכזי בפרויקט זה פיתחנו תוכנית מחשב המממשת את המודל המוצע למציאת מועדי שחרור בפרויקט (כמו גם לפתרון של בעיית RCPSP הבסיסית). פיתוח התוכנית נעשה בשפת התכנות Java ובהתאם לעקרונות אשר למדנו במהלך התואר בקורסים אשר עסקו בעקרונות מדעי המחשב, במבני נתונים ואלגוריתמים. להלן נביא פירוט למושגים הבסיסיים אשר שימשו אותנו במהלך כתיבת התוכנית.

3.4.1. מבני נתונים

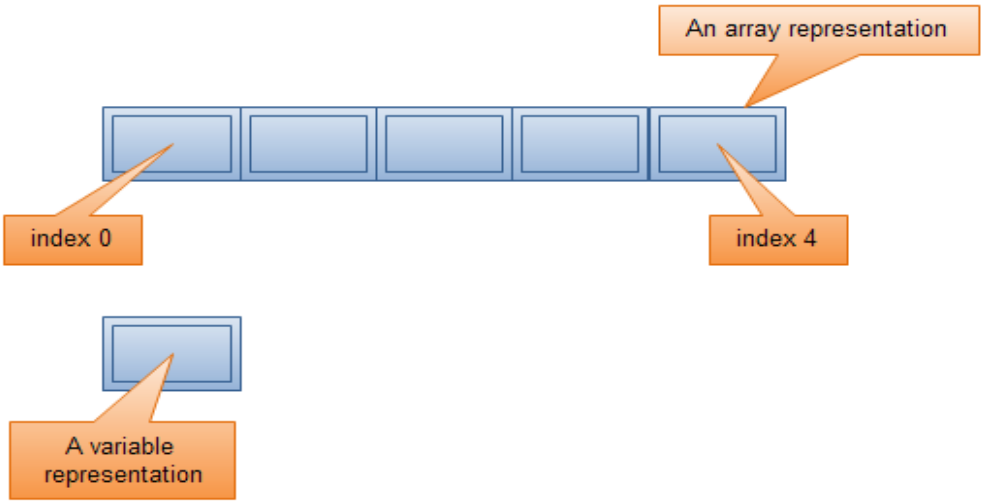
מבנה נתונים הינו דרך מסוימת לאחסון וארגון של נתונים במחשב, כך שניתן יהיה לאתר אותו ולהשתמש בו. סוגים שונים של מבני נתונים מתאימים לצרכים שונים, למשל מבנה נתונים הקרוי "עץ בינארי" מתאים במיוחד למימוש של מסדי נתונים גדולים.

- **המשתנה הפשוט:**

מבנה הנתונים הפשוט ביותר הוא המשתנה הבודד אשר נקרא גם "שדה"; יש לו שם ספציפי (כתובת) והוא יכול להכיל ערך אחד בלבד בכל זמן נתון. כמו כן המשתנה הוא בעל טיפוס מסוים, כלומר הוא יכול להכיל מידע מסוג מסוים בלבד. לצורך הפשטות נציין 3 סוגים עיקריים בלבד: **(1)**תו בודד, **(2)**מספר שלם, ו**(3)**מספר עשרוני. בהקשר של תוכנית מחשב, כל המשתנים המשתתפים בזמן ריצתה מאוחסנים בזיכרון מיוחד שנקרא RAM – Random Access Memory. הוא נקרא כך מכיוון שהוא מאפשר לגשת בצורה ישירה (אקראית) לכל משתנה מבלי לסרוק את כל המשתנים הקיימים, ובלבד שידועה כתובתו. כל מבני הנתונים, מורכבים ככל שיהיו, מורכבים בסופו של דבר ממשתנים בודדים אולם בדרך כלל, אנו משלמים באובדן הגישה האקראית (Random Access) לטובת סדר, ארגון וקומפקטיות של התוכנית.

- **מערך:**

המערך הינו מבנה נתונים המורכב ממספר קבוע (ומצוין מראש) של משתנים הנקראים "תאים", והם ממוספרים בד"כ מאפס ועד n-1 (n הוא אורך המערך). כל המשתנים במערך הם מאותו הטיפוס ולכל אחד מהם ישנה כתובת המוגדרת מצירוף של שם המערך ומספרו הסידורי בתוך המערך. למשל, ע"מ לגשת לתא הראשון במערך בשם Array נציין את הכתובת הבאה: Array[0].



איור 3.4.1.1 – למעלה: סכמה של מערך. למטה: ייצוג של משתנה פשוט.
המערך מורכב מסדרה של משתנים פשוטים.

- **רשומה (Record)**

הרשומה דומה למערך בכך שהיא מקבצת מספר משתנים פשוטים תחת מטריה אחת. ההבדל ביניהם הוא שברשומה טיפוסי הנתונים של המשתנים לא חייבים להיות זהים וכן הם יכולים להיות בעלי שם עצמאי. דוגמא לרשומה המבטאת מספר פרטים בסיסיים על פעילות בפרויקט:

Activity:

Name

Start_Time

End_Time

בדוגמא הנ"ל, לרשומה עצמה קוראים Activity, וע"מ לגשת לאחד מהמשתנים שהיא מכילה יש לציין את שם הרשומה ולאחר מכן את שם המשתנה המבוקש, למשל: Activity**.**Name (הגישה למשתנים של הרשומה היא גישה אקראית).

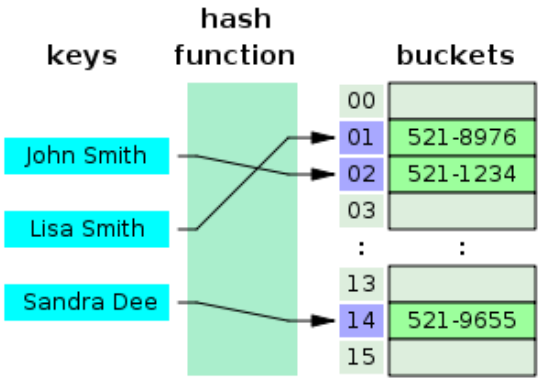
- **רשימה מקושרת (Linked List)**

ברשימה מקושרת נעשה שימוש במבנה הנתונים הפשוט יותר, הרשומה (Record). ניתן "לשדרג" את הרשומה לכדי רשימה מקושרת ע"י הוספת שדה נוסף הנקרא "מצביע" (pointer). תפקידו של המצביע הוא לשמור את הכתובת לרשומה הבאה בתור. בכך הפכנו את הרשומה לחוליה, המסוגלת להתחבר לחוליות נוספות ובכך ליצור שרשרת, או רשימה מקושרת. היתרון הגדול של הרשימה המקושרת הוא שניתן להוסיף ולהחסיר ממנה חוליות בקלות יחסית (בהינתן המיקום המבוקש בו מעוניינים להוסיף או להחסיר), בעוד שבמערך בלתי אפשרי להוסיף/להחסיר אלמנטים (למעט במקרים מסוימים החורגים מהדיון). בעזרת הוספה של מצביע **נוסף** לכל חוליה, אשר ישמור את כתובת החוליה **הקודמת** לה, ניתן אף להאיץ את תהליך ההוספה/הסרה במחיר של זיכרון נוסף כמובן. החיסרון הגדול של הרשימה

המקושרת, כפי שנראה בהמשך, הוא אובדן היכולת לגשת בצורה אקראית לכל אחת מהחוליות שהיא מכילה.

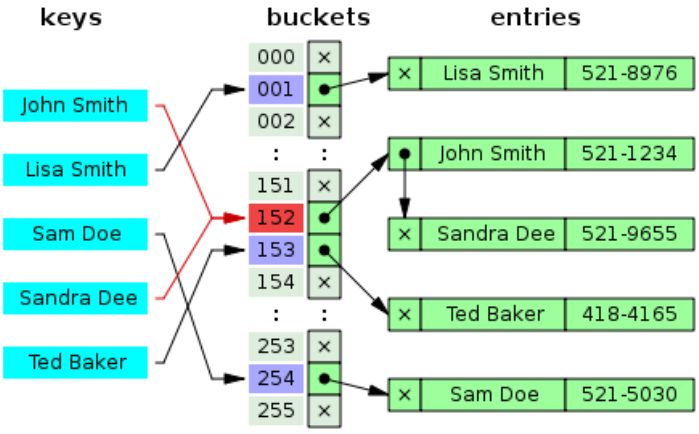
- **טבלת גיבוב (Hash Table)**

טבלת הגיבוב משמשת למיפוי של ערכים לתוך מערך של "דליים". טבלת הגיבוב שואבת את שמה מפונקצית הגיבוב, אשר באמצעותה נעשה המיפוי של הערכים. טבלת הגיבוב היא למעשה מערך, כך שכל תא בו נקרא דלי (bucket). בהינתן ערך חדש אותו מעוניינים להכניס למערך, מפעילים עליו את פונקצית הגיבוב ע"מ לקבל את כתובת הדלי שאליו הוא יכנס. אין זה רלוונטי כל כך כיצד פונקצית הגיבוב ממומשת אך בכל המקרים השאיפה היא שההקצאה לדליים תהיה **פסאודו**-רנדומלית אחידה, עד כמה שניתן. הסיבה לכך היא שאנו מעוניינים לפזר את הערכים בצורה אחידה ככל הניתן על פני המערך עצמו כך שבכל דלי יהיה מספר מינימאלי של ערכים. במערך לא יתכנו מספר ערכים באותו תא ועל כן כאשר שני ערכים או יותר מקבלים את אותו הערך של פונקצית הגיבוב נאמר כי חלה "התנגשות".



איור 3.4.1.2 – סכמה של טבלת גיבוב. מתוך Wikipedia, ערך: Hash table.

נשים לב שהתנגשויות הן בלתי נמנעות אם גודל טבלת הגיבוב (אורך המערך) קטן יותר ממספר הערכים שאמורים להיכנס אליו. זהו בדרך כלל המצב, ומכאן הדליים קיבלו את שמם משום שהם מכילים יותר מאלמנט אחד בו-זמנית. טכניקה מקובלת לפתרון של התנגשויות היא שרשור הערכים ברשימה מקושרת, כך שעבור כל תא במערך המקורי יש רשימה מקושרת משל עצמו.



איור 3.4.1.3 – פתרון התנגשות באמצעות שרשור. מתוך Wikipedia, ערך: Hash table.

3.4.2. בו זמניות (Concurrency)

בו-זמניות הינו מושג במדעי המחשב הבא לתאר תכונה של מערכת (חומרה + תוכנה) המאפשרת לה להריץ חישובים שונים, או יישומים שונים, בעת ובעונה אחת. ביום יום אנו נתקלים תדיר בתכונה הזו בעת שימוש במחשבים אישיים וטלפונים חכמים. ניקח לדוגמא אדם אשר כותב מסמך במעבד תמלילים, בזמן שהוא משתמש בדפדפן אינטרנט ע"מ לחפש תרגום למילה מסוימת, בזמן שהוא מאזין לשיר המתנגן מהמחשב. לכל אחד מהיישומים הללו יכולים להיות תת-יישומים משל עצמו למשל: דפדפן האינטרנט יכול להיות עסוק בלהציג תמונה מהאתר הנוכחי שהמשתמש צופה בו, לנהל הורדה של קובץ גדול וכל זאת בזמן שהוא בודק ללא הרף האם המשתמש הקיש על לחצן כלשהו. ישנם שני סוגים של בו-זמניות: אמיתית ולכאורה. ע"מ להבין את ההבדל ביניהן יש צורך להכיר בכך שכל החישובים מתבצעים בסופו של דבר ע"י המעבד של המחשב (CPU). בד"כ ישנו רק מעבד אחד, אך יכולות להיות לו מספר ליבות. בכל מקרה, ישנם הרבה יותר חישובים ברקע מאשר ישנם מעבדים/ליבות. במחשב בעל מעבד יחיד, ישנה רק משימה אחת (task) אשר ניתן לומר עליה כי היא רצה בנקודה כלשהי בזמן. אם כך, על המשימות השונות להיות מעובדות לפי תור. כאשר המעבר בין המשימות הינו מספיק מהיר, נוצרת אשליה של בו-זמניות. מן הצד השני, מחשבים בעלי מעבדים מכופלי ליבות יכולים להנות מבו-זמניות אמיתית: מספר משימות יכולות להיות מעובדות באותה נקודת זמן, כל אחת על ליבה אחרת (מספר המשימות שרצות באופן בו-זמני אמיתי הוא כמספר הליבות).

4. המודל המוצע לפתרון בעיית תזמון מועדי השחרור בפרויקט

4.1. תיאור של בעיית RDTP (Release Dates Timing Problem)

4.1.1. הקדמה

בפרויקט זה בחרנו להתמקד בשאלה היכן יש למקם את אבני הדרך מסוג "מועדי שחרור" ("Release dates") על פני ציר הזמן של הפרויקט. ע"פ הגדרתן, אבני דרך הן סמן מיוחד על ציר הזמן או אירוע ייחודי על רצף הפעילויות בפרויקט אשר אמור לקבל תשומת לב מיוחדת. בד"כ נהוג להניח אבן דרך בסיומן, או לקראת סיומן, של חבילות עבודה מרכזיות בפרויקט. מלבד תפקידן לאיתות על השלמת רכיב חשוב ובקרה על התקדמות תקינה של הפרויקט, אבני הדרך עשויות גם להעיד על כך שנדרשת החלטה חשובה באותה העת או שמידע בעל חשיבות מיוחדת שעשוי להשפיע על המשך הפרויקט, ולא היה נגיש לפני כן, כעת הפך לנגיש. לפיכך, להחלטה על מיקומן של אבני דרך יכולה להיות השפעה גדולה על ערך הפרויקט. ניתן לייחס לאבני הדרך השפעה ישירה אף יותר אם מביאים בחשבון **מועדי שחרור גרסה או שחרור תכולה כסוג ספציפי של אבני דרך**. תפקידן של אבני דרך מן הסוג הזה הוא לתחום סיומן של תכולות בפרקי זמן מוגדרים מראש בכוונה לשחררן במועד זה לטובת הלקוח, או אף לטובתו של הארגון המבצע עצמו, וכך לקבל עליהן ערך עוד בטרם סיום הפרויקט. דוגמא לפרויקטים המשופעים באבני דרך שכאלה הם פרויקטי (תחזוקת) תוכנה אשר משחררים גרסאות אחת לתקופה לשימוש הלקוחות. הנחת העבודה היא שכל ערך המתקבל כתוצאה מתכולה שהושלמה, מתקבל אך ורק במועד השחרור הקרוב הבא. על כן, יש חשיבות כלכלית לקביעת מועדי השחרור, מעבר לחשיבות האסטרטגית שלהן כאבני דרך.

4.1.2. בעיית RDTP הכללית

הבעיה העומדת בפני מנהל הפרויקט בשלב תכנון הפרויקט הינה כמה מועדי שחרור יש לקבוע, והיכן למקם אותם על פני ציר הזמן. בעיה זו אנו מכנים בשם RDTP – Release Dates Timing Problem. יש לזכור כי **במקביל** יש גם לקבוע את תזמון הפעילויות, זוהי כאמור בעיית ה-RCPSP. אנו מניחים את הנחות העבודה הבאות:

- תזמון הפעילויות מכוון להשגת מטרת השאת הענ"ן של הפרויקט (ולא למשל קיצור משך הזמן).
- **גם** תזמון מועדי השחרור מכוונים לעבר השאת הענ"ן של הפרויקט.
- מס' מועדי השחרור הינו שיקול ניהולי-אסטרטגי והוא נתון מראש*.

**בד"כ בין 2 ל-4 לשנה או למשך הפרויקט. נאמר רק שככל שיש יותר מועדי שחרור, וככל שהערך מתקבל בהרבה נקודות זמן אשר קרובות אחת לשנייה, כך בעיית ה-RDTP הולכת ונעלמת. אולם ריבוי מועדי שחרור עשוי להוביל לתחושה של עבודה בפולסים מהירים וקצרים, עלול להיות בלתי יעיל, לגרום לשחיקת העובדים וליצירת מעמסה על הלקוח שכמובן צריך לבדוק כל תכולה שמשתחררת.*

בהינתן ההנחות הנ"ל, הרי ששתי הבעיות הופכות להיות מסובכות אחת בשנייה (Entangled), פתרון של אחת משפיע על פתרון של השנייה ולהיפך. ע"מ לקבוע את התזמון הכלכלי ביותר של מועדי השחרור יש לדעת מהו התזמון של הפעילויות, מתי הן מתחילות ומתי הן מסתיימות. מן העבר השני, ע"מ לקבוע באופן כלכלי את תזמון הפעילויות יש לדעת מתי מועדי השחרור. את התסבוכת הזו (Entanglement) אנו מכנים בשם "בעיית RDTP **הכללית**".

4.1.3. בעיית RDTP המצומצמת

גישה אחת של פתרון לבעיה הכללית יכולה להיות הוספת N ממדים לבעיית RCPSP המקורית, כך ש-N מייצג את מס' השחרורים ומשתני ההחלטה שנוספים הם זמני השחרור עצמם. חשוב לציין כי מהלך זה מגדיל פי N את מרחב הפתרונות המקורי. לאחר מכן, ניתן להפעיל אלגוריתם מטה-היוריסטי על הבעיה החדשה. ע"מ שהאלגוריתם ישאף לפיזור אחיד של מועדי שחרור ניתן גם לקבוע "קנס" על כל יום חריגה ממשך סטנדרטי של גרסה, כך שמשך סטנדרטי ייקבע כמשך הפרויקט לחלק למס' הגרסאות (N). כחלק מהפתרון המוצע על ידינו לבעיה זו, אני מציעים גישה מתונה יותר, שאינה כרוכה בהכפלה של מרחב הפתרונות ב-N. **כחלק** מהפתרון, אנו מבקשים להסתכל על מצב עניינים פשוט יותר, בו אנו מניחים כי הפעילויות כבר תוזמנו מבעוד מועד וכל שנותר לקבוע הוא מועדם של שחרורי התכולה (כאמור, מספרם של המועדים הוא נתון). את בעיית ה-RDTP במצב זה אנו מכנים בשם "בעיית RDTP **המצומצמת**".

4.1.4. ניסוח מתמטי של בעיית RDTP המצומצמת

בעיה זו "מתלבשת" על בעיית ה-RCPSP שהוצגה בסקירת הספרות וכל הסימונים שהוגדרו בסעיף זה תקפים גם כאן למעט תוספות/השינויים הבאים:

הנחות:

- תזמון הפעילויות (מועדי תחילה וסיום) נתון.
- זרם המזומנים הנובע מסיום פעילות מתחיל אך ורק במועד השחרור הקרוב שלאחר סיום הפעילות.
- מעוניינים במשך אחיד, ככל שניתן, לכל הגרסאות.

נגדיר את הפרמטרים:

N – מספר מועדי השחרור (מספר הגרסאות לשחרור).

$STD.length$ - המשך התקני של גרסה אחת. נקבע כמשך הפרויקט לחלק למספר הגרסאות: t_e/N.

$RD = \{rd_1, rd_2, \ldots, rd_N\}$ - וקטור מועדי השחרור (זמני סיום הגרסאות). **אלו הם משתני ההחלטה**. $(rd_q > 0)$. כברירת מחדל כל גרסה הינה באורך $STD.length$.

$ACT.length_q$ - המשך בפועל של גרסה q (פרמטר מחושב).

$$ACT.length_q = \begin{cases} rd_q - 0 & ,q = 1 \\ rd_q - rd_{q-1}, & ,2 \le q \le N \end{cases}$$

Penalty – גובה הקנס לכל יום חריגה **מעבר*** למשך סטנדרטי של גרסה.

אין טעם לקבוע בנוסף גם קנס לחריגה על משך **קצר ממשך סטנדרטי, משום שכל הארכה של גרסה בהכרח תבוא על חשבון קיצור גרסה אחרת (אחת לפחות). לכן על הקנס הניתן על הארכה להתחשב בעובדה זו.*

pen_q - קנס עבור כל מועד סיום (פרמטר מחושב).

$$pen_q = \max\{\text{ACT.length}_q - STD.length,\ 0\} \cdot penalty \ .$$

$inRD_{i,q}$ - אינדיקאטור האם פעילות i מסתיימת בגרסה נתונה (פרמטר מחושב).

$$inRD_{i,q} = 1 \Leftrightarrow t_i + d_i \le rd_q \ . \ \text{אחרת } inRD_{i,q} = 0 \ . \ \left(1 \le q \le N\right)$$

NPV_i - ערך נוכחי נקי של פעילות i מהוון לזמן t=0. ההשקעה מתבצעת בתחילת הפעילות ואילו תזרים המזומנים מתחיל רק במועד השחרור: $NPV_i = \frac{-I_i}{(1+\alpha)^{t_i}} + \sum_{q=1}^{N} inRD_{i,q} \cdot c_i \cdot \frac{1-(1+\alpha)^{-f_i}}{\alpha \cdot (1+\alpha)^{rd_q}}$

הערה: ההערה בפרק 3.1.3 לגבי חישוב הענ"ן תקפה גם במקרה זה.

ננסח את פונקציית המטרה:

(1) $Maximize \ \sum_{i=1}^{A} NPV_i - \sum_{q=1}^{N} pen_q$

Subject to:

(2) $rd_1 < rd_2 < \cdots < rd_N$

(3) $rd_N = t_e$

ביטוי (1) מחשב את ענ"ן הפרויקט תוך הפחתה של קנסות על משכי גרסאות מעבר למשך הסטנדרטי.

ביטוי (2) שומר על הסדר התקין של מועדי השחרור, כך שהראשון יהיה קודם לשני וכך הלאה.

ביטוי (3) מאלץ את מועד השחרור האחרון להיות מועד סיום הפרויקט.

4.2. פתרון בעיית RDTP הכללית באמצעות פתרון של בעיית RDTP המצומצמת

לאחר שיש בידינו הגדרה של הבעיה המצומצמת, נוכל לפתור (באופן מקורב לפחות) את הבעיה הכללית בשיטה הבאה:

א. נפתור את בעיית ה-RCPSP באמצעות אלגוריתם חמדן כלשהו וכך נקבל אומדן למשך הפרויקט.

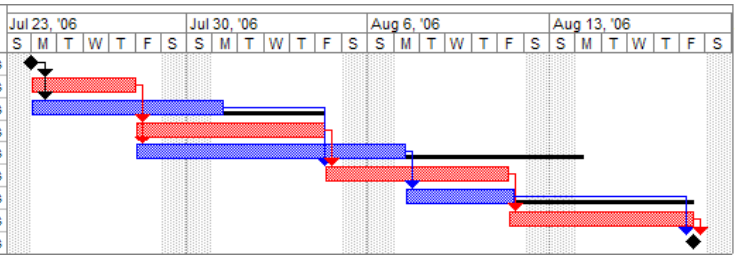
ב. נחלק את אומדן משך הפרויקט במספר מועדי השחרור הרצוי וכך נקבל את מועדי השחרור **המשוערים**.

ג. על בסיס מועדי השחרור המשוערים, נפעיל אלגוריתם מטה-היוריסטי לפתרון **מחדש** של RCPSP למציאת תזמון הפעילויות.

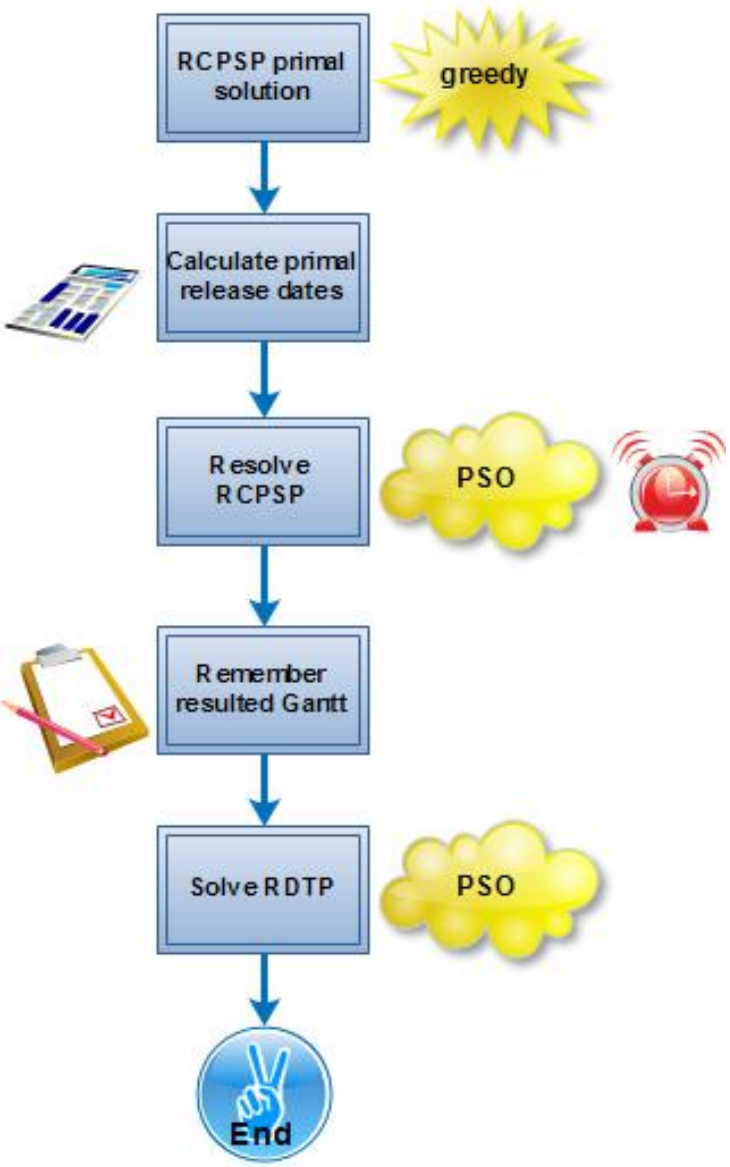
ד. על בסיס תזמון הפעילויות **החדש** שהתקבל, נפעיל אלגוריתם מטה-היוריסטי לקביעה מחדש של מועדי השחרור ע"י פתרון בעיית RDTP המצומצמת.

הרציונל: לבעיית RCPSP יש A מימדים (A = מס' הפעילויות) ואילו לבעיית RDTP המצומצמת יש N מימדים (N = מס' מועדי השחרור). מכיוון ש $A >> N$, מירב הסיבוכיות וזמן הריצה נובעים מהפעלה של פונקצית המטרה של בעיית RCPSP. נרצה לייצר פתרון טוב ככל שניתן אך במאמץ קטן ככל שניתן ע"י הימנעות מהגדלת הסיבוכיות. באמצעות השיטה הנ"ל אנו נמנעים מהכפלת מרחב הפתרונות הגדול ממילא של RCPSP ב-N וכך נמנעים מתוספת סיבוכיות לבעיה ומצליחים לשמור על פתרון יעיל ומהיר. נשים לב שאע"פ שבשיטה המוצעת אנו פותרים את RCPSP פעמיים, הרי שבסעיף א' אנו מפעילים את פונקצית המטרה פעם אחת בלבד ועל כן עלות סעיף זה מבחינת זמן ריצה הינה זניחה.

ID	Task Name	Predecessors	Duration
1	Start		0 days
2	a	1	4 days
3	b	1	5.33 days
4	c	2	5.17 days
5	d	2	6.33 days
6	e	3,4	5.17 days
7	f	5	4.5 days
8	g	6	5.17 days
9	Finish	7,8	0 days



איור 4.2.1 – תרשים גאנט. הבעיה היא לקבוע את מועדי השחרור. מתוך Wikipedia, ערך: Gantt chart.
נוצר באמצעות Microsoft Project.



איור 4.2.3 – סכמה המתארת את הפתרון המוצע למציאת מועדי השחרור.

השעון האדום מסמן את השלב המסוכן ביותר מבחינת ביצועים.

4.3. חיזוק כושר אלגוריתם PSO

בפרק זה אנו מציגים את מושג "כושר האלגוריתם". כושר האלגוריתם הינו היכולת של האלגוריתם למצוא פתרונות טובים, כך שפתרון טוב נמדד ביחס לפתרון האופטימאלי לבעיה. ככל שכושר האלגוריתם גבוה יותר, כך הוא יניב פתרונות טובים יותר ועבור מספר רב יותר של הפעלות. לדוגמה: אלגוריתם הקולע בול לפתרון האופטימאלי של הבעיה בכל הפעלה ה-100, אך בשאר ההפעלות מניב פתרון רחוק מאוד מהפתרון האופטימאלי – נאמר עליו שהוא בעל כושר נמוך. ע"מ לחזק את הכושר של האלגוריתם נקטנו במספר פעולות משמעותיות:

א. בחירה חכמה של המשקולות לרכיבי המהירות: הנטייה ההתחלתית שלנו הייתה לתת משקל זהה לכל רכיבי המהירות, וכן שסך כל המשקולות יסתכם ב-1 : $\omega = \varphi_g = \varphi_p = 0.33$. יש לציין כי הרצון שסך המשקולות יהיה שווה לאחד הינו בעיקר קוסמטי ונועד עבור המשתמש להבחין ביתר קלות ביחסים שבין משקלם של המרכיבים השונים. לגבי המשקולות עצמן, ניסיוננו לימד אותנו כי הצירוף המיטבי הוא דווקא הצירוף הבא: $\omega = 0.2,\ \ \varphi_g = 1,\ \ \varphi_p = 0.3$. הרציונל: ראשית, כוחו ושמו של האלגוריתם נובע מאפקט הנחיל שבו כל החלקיקים *אמורים* לנוע, בסופו של דבר, לאותו אזור של פתרונות. אילו המשקל הניתן לכיוון הפתרון הטוב הגלובאלי הינו נמוך מסכום המשקולות האחרים, הרי שהחלקיקים יטו יותר לעבר אזורים אחרים, כל חלקיק והנטייה שלו, ואפקט הנחיל ישבר. מצד שני, ערך גבוה של φ_g יגרור התכנסות מהירה של החלקיקים, דבר שברוב המקרים מתבטא ב"נפילה למלכודות" של אופטימום מקומי. על כן יש לקבוע את φ_g על רמה נמוכה מספיק מבלי לשבור את אפקט הנחיל.

באשר למשקל הניתן לכיוון הפתרון הטוב האישי, הרי שגם הוא צריך להיות גבוה ע"מ שהחלקיק יימנע מלעזוב אזורים מבטיחים. במקביל, יש לשמור על אינרציה בלתי-זניחה ע"מ לאפשר לחלקיק לפגוש במקרה פתרונות עדיפים. כמו כן האינרציה שימושית כאשר הערך המיטבי הנוכחי נמצא על צלע של "הר" ואנו מעוניינים שהחלקיק ימשיך ו"יטפס" למעלה.

ב. איזון רנדומאליות: בהמשך ישיר לטיעונים שהועלו בסעיף א', בחרנו למזער את השפעתם של האלמנטים הרנדומאליים בנוסחת עדכון המהירות. הסיבה לכך היא שהאלמנטים הרנדומאליים מעוותים את יחס הזהב שתואר לעיל, מה שמחבל בהתכנסות של החלקיקים ומביא לפתרונות נחותים יותר. מצד שני, לא ניתן להתווכח עם היתרון שמספקת הרנדומאליות במציאת פתרונות עדיפים "במקרה". הצלחנו להשיג את האיזון המבוקש ע"י התחשבות באלמנטים הרנדומאליים בכל איטרציה שנייה, ואך ורק בהן.

ג. נרמול הרכיבים של וקטור המהירות: פעולה משמעותית ביותר שנקטנו בה היא נורמליזציה של כל אחד מהרכיבים של וקטור המהירות. לאלגוריתם ה-PSO המקורי יש שני חסרונות גדולים. שני החסרונות נובעים מהעובדה שככל שחלקיק רחוק יותר מנקודת ערך מיטבי גלובאלי/אישי כך רכיב המהירות בכיוון זה הינו גדול יותר. החיסרון הראשון הינו אפקט מטוטלת שעלול להיווצר בין שני

הערכים המיטביים הנ"ל. ככל שהוא מתקרב לאחד כך הוא נמשך לשני, ובסופו של דבר עלול החלקיק להילכד בתווך שבין שתי הנקודות הללו מה שנוגד את המטרה להשיג את אפקט הנחיל וכמובן פוגע בתוצאות המתקבלות. החיסרון השני הינו קיומם של מצבים בהם החלקיק מדלג על מרחב פתרונות גדול מדי. זאת קורה כאשר החלקיק רחוק מנקודת הערך המיטבי ולכן צובר מהירות גבוהה. השם "מהירות" הינו מטעה כיוון שהחלקיקים אינם באמת מבקרים בכל הנקודות בין המוצא ליעד שלהם, אלא הם "עפים" לשם (השימוש במושג עפים, fly , מקובל גם בספרות), ולכן מהירות גבוהה מדי אינה רצויה. נרמול של וקטור המהירות בכל איטרציה (כולל בשלב אתחול המהירות) גורם לחלקיק לקחת צעד אחד מדוד בכל פעם מה שמבטל את שני החסרונות הנ"ל. נציין שהנרמול עולה בחישובים נוספים כמובן, אך המטרה האולטימטיבית של האלגוריתם הינה אספקת תוצאות הקרובות לאופטימום הגלובאלי עד כמה שניתן, ולכן התועלת עולה על ההפסד המתבטא בתוספת זמן ריצה (הסיבוכיות נשארת זהה). בפרק 5 נקיים דיון רחב יותר בנושא סיבוכיות האלגוריתם זמן הריצה.

ד. **צעד הולך וקטן**: פעולה זו הינה שיפור לפעולה שהוצגה בסעיף ג' ובמובן מסוים גם תיקון שלה. למרות זאת, בחרנו להציג אותה כפעולה עצמאית כיוון שהפעולה המוצגת בסעיף ג' אפקטיבית דיה בפני עצמה. כאשר החלקיקים מתקרבים לנקודת ההתכנסות גובר הסיכון שהם ידלגו מעל הערך המיטבי, זאת מכיוון שנרמלנו את הצעד שהם מבצעים בכל פעם. הדבר מתבטא בהתכנסות בלתי מושלמת של החלקיקים ומונע איתור של פתרונות עדיפים הנמצאים קרוב מאוד לפתרון המיטבי הנוכחי. כדי להתגבר על החיסרון הזה מימשנו מנגנון אשר מקטין את הצעד המנורמל ביחס ישר למספר האיטרציות, כך שבתחילת הריצה הצעדים הם גדולים ואילו לקראת סוף הריצה הם שואפים לאפס.

ה. **שכפול מרחב הפתרונות**: מרחב הפתרונות הינו המרחב שבו נע החלקיק והוא תחום בין וקטורי החסמים העליון והתחתון. בעיות התזמון הינן בעיות קומבינטוריות בעלות מרחב פתרונות סופי וניתן לקבוע את כולו כמרחב בו מתמרנים החלקיקים. אנו מצאנו כי עוזר לשכפל את מרחב הפתרונות משום שכך יש לחלקיק סיכוי כפול לפגוש בדרכו פתרונות עדיפים. כפי שיוצג בפרק הבא, אנו משתמשים בפונקצית הסינוס כדי להמיר את מיקום החלקיק לתזמון תקין. על כן, קבענו את מרחב הפתרונות להיות בין 0 ל- 4π בכל מימד (במקום בין 0 ל- 2π).

ו. **ערך הולך וגדל של משקולת** φ_p : מסקירת הספרות בנושא אלגוריתמי PSO עלה כי הגדלת המשקולת של מיקום הערך המיטבי האישי עוזרת בהגברת כושר האלגוריתם. במקרה שלנו, טכניקה זו עוזרת לנו להימנע ממספר גבוה מדי של איטרציות. נסביר מדוע: כיוון שנרמלנו את הצעדים של כל חלקיק, ואף קבענו כי הצעדים הולכים וקטנים, הרי שלחלקיקים דרוש זמן רב יותר (=יותר איטרציות) ע"מ להתכנס. בכך שאנו מתחילים עם משקל מינימאלי של הערך המיטבי האישי אנו גורמים לחלקיקים

להתכנס מהר יותר לערך המיטבי הגלובאלי ובכך מפצים על האיטיות הנגרמת מסעיפים ג' + ד'. משקולת הערך המיטבי האישי גדלה בהדרגה עד שהיא מגיעה לערך של φ_p שנקבע ע"י המיישם.

4.4. התמודדות עם בעיית מרחב הפתרונות הבדיד

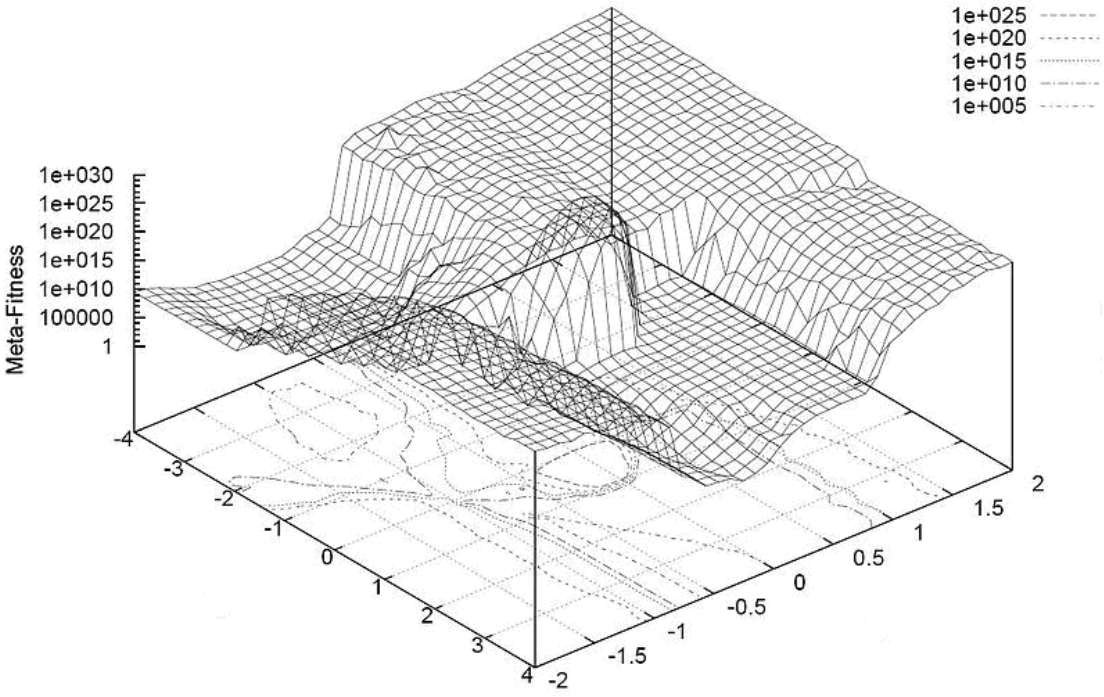
אלגוריתם PSO המקורי תוכנן לפתור בעיות עם מרחב פתרונות רציף ואין הוא מסוגל להתמודד עם בעיות קומבינטוריות אשר להן מרחב פתרונות בדיד. ישנן שתי אפשרויות בבואנו לגשר על הפער:

א. ביצוע שינויים **באלגוריתם** עצמו כך שיוכל להתמודד עם בעיות בעלות מרחב פתרונות בדיד.

ב. ביצוע שינויים **בפונק' המטרה** כך שמרחב הפתרונות שלה יהיה רציף.

במקורות [1] ו-[2], ניתן למצוא הצעות לשינויים באלגוריתם PSO עצמו (אומנם לבעיית ה-JSP, אך ניתן לראות בה מקרה פרטי לבעיית ה-RCPSP). אנו מוצאים את השינויים הללו מסורבלים לשווא וחוטאים למטרה שכן השינויים הם כה גדולים עד שאין טעם עוד לקרוא לתוצר הסופי בשם אלגוריתם PSO. בעבודה זו אנו נוקטים בגישה ב' אך במקביל מציעים לשמור על הכושר של האלגוריתם המקורי ע"י הכנסת שינוי מינורי שאינו פוגע בפשטות והאלגנטיות שלו.

השיטה שאנו מציעים היא "לעטוף" את הבעיה המקורית בשכבה אשר תמיר פתרונות ממרחב רציף לפתרונות במרחב בדיד. על השכבה לקבל ווקטור של מספרים ממשיים כלשהם ולהמירם לתזמון **תקין** (כלומר לא מסתפקים בהפיכת המספרים הממשיים למספרים שלמים).



איור 4.4.1 – המחשה של מרחב פתרונות רציף תלת מימדי של פונקציה כלשהי.

מתוך Wikipedia, ערך: Particle Swarm Optimization.

4.4.1. עטיפה לבעיית RCPSP

נגדיר תזמון לבעיית RCPSP כך שיבוטא באמצעות תמורה של הפעילויות: $\pi = \{\pi_1, \pi_2, ..., \pi_n\}$, כך שהפעילות π_1 היא הראשונה לעלות על הגאנט, לאחר מכן פעילות π_2 וכך הלאה.

התנאים הנדרשים לתקינות התזמון הינם כפי שפורטו בתיאור הבעיה בסקירת הספרות.

היות ותזמון תקין מכיל A פעילויות (לא כולל פעילויות ה-start וה-end), הרי שמכאן נגזר כי לבעיית RCPSP יש A מימדים. למעשה ניתן לומר כי ישנם A-1 מימדים כיוון שהשיבוץ האחרון הוא שיבוץ הנגזר מכל קודמיו, אך אין בעיה מבחינה מתמטית עם ההגדרה של A מימדים ולכן נדבוק בהגדרה זו למען הפשטות. נרצה להמיר וקטור מספרים ממשיים באורך A לתזמון תקין. נזכיר שבהקשר של אלגוריתם PSO אותו וקטור שנסמן כ-$X = \{x_1, x_2, \ldots, x_i, \ldots, x_A\}$, הוא בעצם מערך הקואורדינאטות המבטאות מיקום של החלקיק. ברור לחלוטין שעיגול פשוט של כל ערך בנפרד אומנם יניב ווקטור של ערכים שלמים אך לא יספק תזמון שיעמוד בתנאי הבעיה. אנו מציעים את העטיפה הבאה:

שלב א':

צעד א': הגדר פעילויות נעולות ופעילויות פתוחות. פעילות פתוחה הינה פעילות שטרם תוזמנה אך כל הפעילויות המקדימות שלה כבר תוזמנו. פעילויות נעולות הן כל השאר.

צעד ב': צור רשימה של כל הפעילויות הפתוחות לפי סדר האינדקסים שלהן: $1 \to 2 \to \ldots \to A$.

שלב ב':

עבור כל ערך x_i בווקטור המיקום של החלקיק בצע:

צעד א': חשב את הסינוס של הערך: $x_i \leftarrow \sin(x_i)$. צעד זה נועד ליצור מחזוריות ע"מ להתמודד עם מצבים בהם החלקיקים עוזבים את אזור החיפוש שהוגדר. בניגוד לבעיות "קונבנציונליות" בעלות מרחב פתרונות רציף ואינסופי, בעיות קומבינטוריות הן בעלות מרחב פתרונות סופי. כמו כן, המחזוריות מגדילה הסיכוי של חלקיק לפגוש מגוון גדול יותר של פתרונות גם אם הוא ממשיך באותו כיוון יחסי.

צעד ב': נרמל את הערך המתקבל לערך ממשי בין 0 ל-1 . מתבצע כך: $x_i \leftarrow (x_i + 1)/2$.

צעד ג': נרמל את הערך המתקבל לערך שלם בין 1 לאורך רשימת הפעילויות הפתוחות. מתבצע כך: $x_i \leftarrow rounddown(x_i \cdot length) + 1$.

צעד ד': מצא ברשימת הפעילויות הפתוחות את הפעילות במיקום ה- x_i והצב אותה כבאה בתור בווקטור התזמון π (וקטור התוצאה).

צעד ה': מחק את הפעילות ששובצה מרשימת הפעילויות הפתוחות ועדכן את הרשימה בהתאם לאילוצי הקדימויות.

שלב ג':

צעד א': שבץ הפעילויות על הגאנט ע"פ וקטור התזמון שהתקבל, π. כל פעילות תשובץ בזמן המוקדם ביותר בו יכולה להתחיל מבלי להפר את אילוצי הקדימויות והמשאבים.

צעד ב': חשב ענ"ן לכל פעילות בנפרד.

צעד ג': סכום את ערכי ה-NPV של הפעילויות. כך מתקבל הענ"ן של הפרויקט.

4.4.2. עטיפה לבעיית RDTP

לכאורה, בעיית RDTP אינה אמורה לסבול מבעיית מרחב הפתרונות הבדיד כיוון שמשתני ההחלטה שלה הם זמנים, ולכן אמורים להיות רציפים מלכתחילה. עובדה זו הינה נכונה מבחינה תיאורטית אולם נזכור כי אנו מניחים שהזמן הינו בדיד ונמדד בימים שלמים בלבד ועל כן, גם בעיית RDTP זקוקה לעטיפה הממירה פתרונות ממרחב רציף לפתרונות במרחב בדיד. יכולנו להגדיר עטיפה אשר מקבלת כקלט מספר בין 0 ל- 4π וממירה אותו למספר בין 0 ל-1 ולאחר מכן כופלת במשך זמן הפרויקט ומעגלת התוצאה. בשיטה זו היינו מקבלים עטיפה אשר ממירה רצף מספרים עשרוניים לרצף מספרים שלמים, כך שכל אחד מהם מייצג יום כלשהו בחיי הפרויקט. בולט בהיעדרותו הוא מנגנון המבטיח סדר הגיוני של מועדי השחרור, כך שמועד t=32 לא יבוא לפני מועד t=24 למשל. פחות בולט בהיעדרותו הוא מנגנון המונע קבלת מועדי שחרור בזמנים בהם לא מסתיימת אף פעילות. למרות שמבחינה מעשית מנגנון שכזה אינו הכרחי ואין שום בעיה עם מועדי שחרור שכאלה, מצאנו שמנגנון שכזה דווקא עוזר לשפר את כושר האלגוריתם במקרים מסוימים כיוון שהוא מוותר מראש על הרבה פתרונות נחותים. הויתור על הפתרונות הללו בא במחיר של ירידה בכמות ההפרשים האפשריים שבין מועד שחרור לזה שבא אחריו, ועל כן הוא יעיל כל עוד הקנס על חריגה ממשך סטנדרטי של גרסה הוא לא גבוה מאוד ביחס לתקבולים מהפעילויות וכל עוד מספר הפעילויות הינו רב (90 לפחות). בסופו של דבר החלטנו לוותר על המנגנון הזה לטובת התאמה של הכלי גם לפרויקטים קטנים יותר ובכך לשמר את הורסטיליות שלו. השיטה בה בחרנו לממש את העטיפה לבעיית RDTP היא כדלהלן:

שלב א':

צעד א': מצא את מועד הסיום של הפרויקט te.

צעד ב': אתחל מועד שחרור זמני temp←0.

שלב ב':

עבור כל ערך x_i בווקטור המיקום של החלקיק בצע:

צעד א': חשב את הסינוס של הערך: $x_i \leftarrow \sin(x_i)$.

צעד ב': נרמל את הערך המתקבל לערך ממשי בין 0 ל-1. מתבצע כך: $x_i \leftarrow (x_i + 1)/2$.

צעד ג': נרמל את הערך המתקבל לערך שלם בין מועד השחרור האחרון שחושב לבין מועד סיום הפרויקט. מתבצע כך: $x_i \leftarrow \left[rounddown\left(x_i \cdot (t_e - temp)\right)+1\right]+temp$.

צעד ד': הצב המועד שהתקבל בווקטור התזמון RD (וקטור התוצאה) ועדכן את temp.

<u>שלב ג'</u>:

צעד א': שבץ על הגאנט את מועדי השחרור בווקטור RD שהתקבלו.

צעד ב': חשב ענ"ן לכל פעילות בנפרד.

צעד ג': סכום את ערכי ה-NPV של הפעילויות.

צעד ד': חשב הקנס הנובע ממשכי הגרסאות והפחת מן הענ"ן שהתקבל.

4.4.3. שמירה על הכושר של האלגוריתם

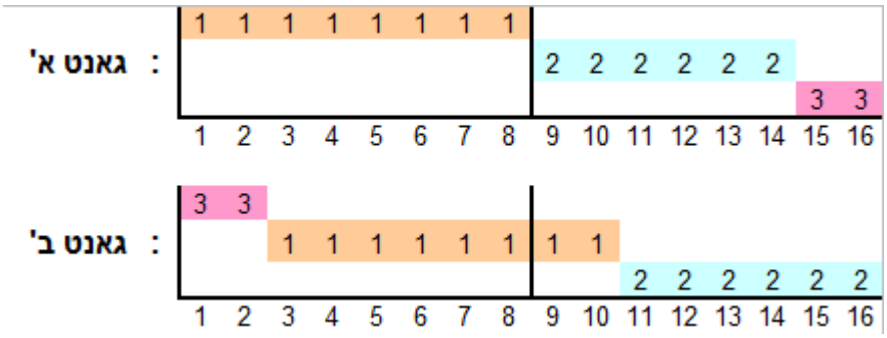
תוצר לוואי של העטיפות הנ"ל הינה העובדה שמספר פתרונות שונים מהמרחב הרציף יתורגמו לאותו פתרון מהמרחב הבדיד ולכן גם לאותה תוצאה סופית (האשמה נופלת בעיקר על פעולת ה-rounddown). עובדה זו כשלעצמה אינה גורעת מהיכולת להפעיל את אלגוריתם PSO המקורי על הבעיה העטופה, אך היא כן פוגעת במידת מה בביצועיו, מבחינת הקירוב לפתרון האופטימאלי. הבעייתיות נעוצה בעובדה שהאלגוריתם מתחזק ערך מיטבי גלובאלי **יחיד**, g_best (ערך פונק' המטרה של g) . מיקום זה משפיע על מהירות כלל החלקיקים, גם כאלו שהערך המיטבי האישי שלהם **זהה** ל-g_best. כתוצאה מכך, גם חלקיקים שנמצאים באזורים מבטיחים עוזבים אותם ונמשכים אל האזור של g, כלומר הסיכון כי החלקיקים ייקלעו לסביבה של אופטימום מקומי גובר. ע"מ להתגבר על הבעיה הזו אנו מציעים להכניס שינוי בחישוב המהירות של החלקיקים. במידה והערך המיטבי של החלקיק זהה לערך המיטבי הגלובאלי, יש להתחשב ב - φ_g כאילו הוא שווה לאפס ומצד שני להגדיל פי 2 את הערך של φ_p. ע"י כך, החלקיקים הללו יתעלמו מ-g לחלוטין וימשיכו לנדוד בכיוון הערך המיטבי האישי שלהם (והאינרציה). בשיטה זו האלגוריתם סוקר מרחבים מגוונים יותר והסיכון להתכנסות לאופטימום מקומי פוחת.

4.5. לשם מה נחוץ לנו MGPSO?

MGPSO הוא חיוני ביותר להשגת תזמון יעיל של מועדי השחרור, בהינתן המודל המוצע. לכאורה, ניתן להתייחס ל-MGPSO כאל עוד טכניקה לשיפור כושר האלגוריתם ע"י הימנעות מנפילה למלכודות אופטימום מקומי הפועלת בשיטת "אם אינך יכול לנצח אותם, הצטרף אליהם". בתיאוריה, אלגוריתם MGPSO יכול להניב מספר תוצאות בריצה בודדת ולכן יכול לחשוף מספר נקודות אופטימום מקומיות במקביל, מהן ניתן לברור את האופטימום הגבוה ביותר. בפועל, יעילותו של MGPSO פוחתת פי n, כאשר n הוא מספר נקודות האופטימום המקומי שאנו מעוניינים לחשוף, ולכן אנו לא משתמשים בו כדי להימנע מהן. נהפוך הוא – אנחנו משתמשים בו ע"מ למצוא דווקא פתרונות קרובים אך נחותים יותר מהפתרון האופטימאלי!

הבעיה:

MGPSO בא לפתור בעיה שורשית באלגוריתם המוצע הנ"ל. האלגוריתם מניח הנחה סמויה שתזמון הפעילויות האופטימאלי עבור בעיית RCPSP, עם מועדי השחרור המשוערים, הוא בהכרח תזמון המוצא האופטימאלי ל-RDTP. זוהי הנחה שגויה! ההנחה שגויה, בין השאר, בגלל העובדה שתזמון הפעילויות נבנה על סמך מועדי שחרור משוערים ולא לפי מועדי השחרור הסטנדרטיים האמיתיים. נוסף על כך, הפתרון לא נבנה על סמך מועדי שחרור עם מרווחים קבועים ביניהם, זאת מכיוון שלא ניתן לדעת מראש את משך הפרויקט. כיוון שכך, מועד השחרור המשוער האחרון עובר תיקון בכל הפעלה של פונקצית המטרה של RCPSP כדי להתאים למועד הסיום של הפרויקט ע"פ התזמון שנבחן באותו הרגע, ורק לאחר מכן מחושב הענ"ן (מועד השחרור האחרון חייב להיות שווה למועד סיום הפרויקט. שאר מועדי השחרור נשארים קבועים לכל אורך הפתרון של RCPSP). אי לכך, נוצרת הטיה בלתי נמנעת במשך הגרסה האחרונה לעומת משכן של שאר הגרסאות והפתרונות המתקבלים מחושבים על סמך ההטיה הזו. אפילו אם היינו יכולים להימנע מההטיה, אנו עדיין ניצבים בפני בעיה: שכן ייתכנו מספר תזמונים *שונים* המניבים ענ"ן זהה או דומה. בהינתן מספר תזמונים שכאלה, צפוי שתהיה שונות ביניהם מבחינת היכולת לשנות את מועדי השחרור, אע"פ שכאמור הענ"ן שלהם הוא דומה. כדי להמחיש את הבעיה נתבונן באיור הבא:



איור 4.5.1 – הדגמה לצורך ב-MGPSO.

שני תרשימי הגאנט שבציור מתארים פרויקט בעל 3 פעילויות. נניח:

- אין אילוצי קדימויות
- שלוש הפעילויות צורכות את מקסימום המשאבים הקיימים ולכן אינן יכולות להתקיים במקביל.
- קיימים 2 מועדי שחרור התחלתיים: אמצע הפרויקט וסיום הפרויקט.
- משך סטנדרטי לגרסה: 8 ימים. קנס לחריגה ליום: 100 ₪ . ריבית: 1% ליום.
- כל פעילות מניבה 7000 ₪ בסיומה. נזניח עלות ההשקעה.

ענ"ן תזמון א' הינו: $\frac{7000}{(1+0.01)^8}+\frac{7000\cdot 2}{(1+0.01)^{16}}=18{,}403.88$

ענ"ן תזמון ב' הינו: $\frac{7000}{(1+0.01)^8}+\frac{7000\cdot 2}{(1+0.01)^{16}}=18{,}403.88$

לשני התזמונים ענ"ן זהה כיוון שבשני המקרים תכולה אחת משתחררת במועד השחרור הראשון ואילו שתי תכולות משתחררות בסיום הפרויקט. אולם, את תזמון ב' קל לשפר ע"י דחייה של מועד השחרור הראשון בשני ימים ע"מ לשחרר שתי תכולות במועד זה.

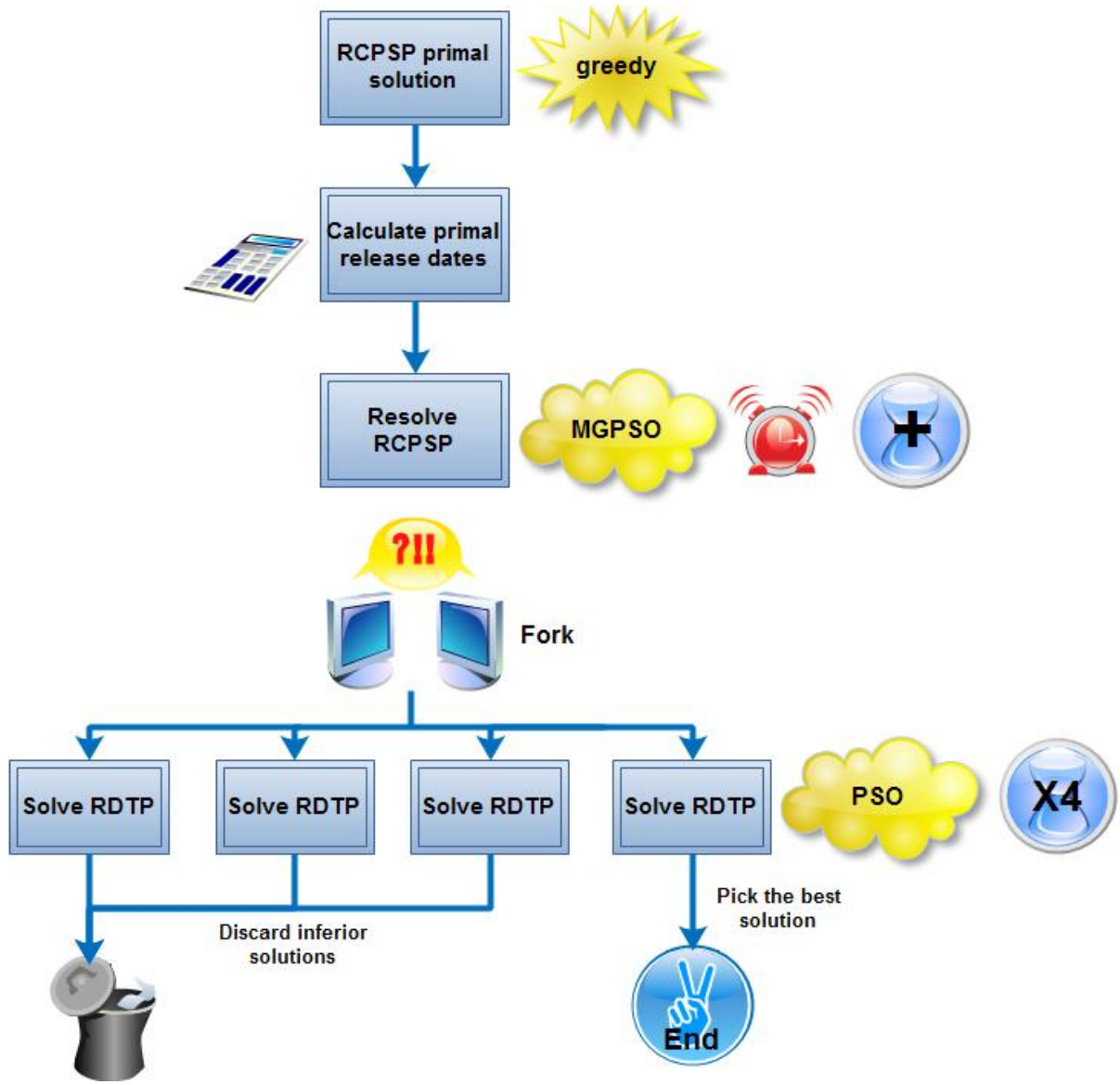
ענ"ן תזמון ב' לאחר דחיית מועד השחרור ביומיים: $\frac{7000\cdot 2}{(1+0.01)^8}+\frac{7000}{(1+0.01)^{16}}-100\cdot 2\approx 18{,}443$

קיבלנו שיפור של 40 ₪. את תזמון א' לא ניתן לשפר בצורה כזו ולמעשה לא ניתן לשפר כלל.

באלגוריתם PSO המקורי אנו יכולים לקבל רק אחד משני התזמונים בכל ריצה נתונה. לעומת זאת, MGPSO נותן לנו את האפשרות לקבל את שני התזמונים יחד, כך שנוכל לבחון את יכולת השיפור של כל אחד מהם ולבחור בתזמון העדיף.

הפתרון:

למרות כל זאת אין באי-נכונותה של ההנחה הנ"ל לפסול את כל האלגוריתם. על ההנחה שבנדון לעבור עידון ולהיות מנוסחת באופן הבא: התזמון המתקבל מפתרון של RCPSP הוא תזמון *מקורב* לתזמון המוצא האופטימאלי של RDTP וזאת מכיוון שהוא *שואף* לרווח בצורה אחידה את מועדי השחרור תוך השאת הענ"ן. מכיוון שאין הכרח שענ"ן התזמון האופטימאלי ל-RCPSP יהיה גבוה או שווה לענ"ן תזמון המוצא האופטימאלי עבור RDTP, הרי שעלינו לסקור גם פתרונות אופטימום מקומיים "נחותים" יותר. בפרויקט זה בחרנו לממש את האלגוריתם כך שיסקור 4 נקודות אופטימום מקומיות באמצעות טכניקת MGPSO.



איור 4.5.2 – סכמה המתארת את הפתרון המוצע המתוקן למציאת מועדי השחרור.

שעוני החול הכחולים מסמלים תוספות בזמן ריצה ביחס לאלגוריתם הראשון.

5. שיפור ביצועים

5.1. כללי

כפי שכבר הצהרנו, מטרת הפרויקט העיקרית הינה פיתוח של כלי לפתרון בעיות RCPSP ו-RDTP בעלות זיקה מרבית למציאות. אחד ההיבטים הדורשים התייחסות אם כך הוא היבט הביצועים של הכלי. במציאות, לזמן יש ערך ומנהלי פרויקטים ומתכנני תזמונים לא יטו להשתמש בכלי אשר גוזל זמן פתרון רב, על אחת כמה וכמה בכלי היוריסטי אשר קרוב לוודאי יצריך מספר הרצות ולאחר מכן השוואה ביניהן. מעבר לפן הכלכלי, לחוויית המשתמש יש חשיבות רבה; סביר שמשתמש יעדיף להשתמש בכלי נוח ומהיר על פני כלי מסורבל ואיטי. בהזדמנות זו נציין שחלק ניכר מאוד מהמאמצים בכתיבת הפרויקט הושקעו בפיתוח הכלי, אשר כוחו יפה גם מחוץ להקשר המידי של שאלת המחקר. בהיבט זה, אנו רואים בפרויקט גם בעל פן מעשי ולא רק מחקרי גרידא.

נזכור שמלכתחילה פנינו לפתרון ה-היוריסטי כיוון שהשיטות המדויקות אינן פתירות עבור בעיות בסדרי גודל גדולים (מציאותיים) משום שהן צורכות זמן עיבוד בלתי הגיוני. מן הצד השני, אלגוריתם היוריסטי, כשלעצמו, מציע לכל היותר פתרון בזמן פולינומיאלי שגם הוא יכול להיות ארוך למדי. האם זה אומר ששיטות מטה-היוריסטיות הן פשרה בין טיב הפתרון למהירות? התשובה היא לא. שיטות מטה-היוריסטיות יכולות להניב פתרונות טובים למדי, אך לא פחות חשוב מכך, במהירות גבוהה מאוד.

לצורך הדיון, נבצע הבחנה בין מושג "כושר האלגוריתם", עליו כבר דיברנו בפרקים קודמים, לבין מושג "ביצועי האלגוריתם". כושר האלגוריתם הינו מושג המתייחס לאלגוריתם כאל ישות **תיאורטית** בלבד ומתייחס ליכולת שלו להגיע לפתרונות טובים, כאשר פתרון טוב נמדד בקרבה שלו אל הפתרון האופטימאלי. ע"פ הגדרה זו, שיטות מדויקות, המבוססות על "הסתעף וחסום" למשל, הן בעלות כושר גבוה במיוחד, משום שהן תמיד יניבו פתרון אופטימאלי. נשים לב כי אין כלל התייחסות למהירות ההתכנסות לפתרון. מאידך, מושג הביצועים מתייחס **לדרך המימוש** של האלגוריתם. עבור אלגוריתם נתון, יכולות להיות דרכים רבות לממש אותו: בשפות שונות, בטכניקות תכנות שונות וכו'. מושג ביצועי האלגוריתם אם כן, מתייחס לא לאלגוריתם התיאורטי אלא לאלגוריתם הכתוב בפועל, בשפת התכנות. אלגוריתם בעל ביצועים טובים הוא אלגוריתם יעיל. לצורך פישוט הדיון, נצמצם את מושג היעילות לכדי מהירות הריצה של האלגוריתם (הזמן שלוקח לו לסיים). הכלי שפיתחנו בפרויקט זה הינו בעל ביצועים מצוינים המאפשרים לו לפתור בעיות גדולות, המכילות 120 פעילויות ואף למעלה מזה, במספר שניות בודדות. בפרק זה נפרט על שתי טכניקות עיקריות בהן נקטנו ע"מ להגיע לביצועים הללו.

5.2. ההשפעה של MGPSO על יעילות האלגוריתם למציאת מועדי השחרור

החיזוק המשמעותי שמביא איתו MGPSO אינו בא בחינם אלא הוא דורש חישובים נוספים. עם זאת, נראה שהעלות מצדיקה את התמורה. ניתן לחלק את ההשפעה של הכנסת השימוש ב- MGPSO לשני חלקים: **(א)**תוספת החישובים שנדרשים ע"מ לשדרג את אלגוריתם ה-PSO ו-**(ב)**תוספת החישובים להם אנו נזקקים כדי לטפל בפתרונות שנוספים. ראשית, יש להזכיר כי ע"מ למצוא את מועדי השחרור האופטימאליים אנו משתמשים ב-PSO פעמיים; פעם אחת למציאת פתרון ל-RCPSP ופעם שנייה

למציאת פתרון ל-RDTP (המצומצמת). ע"פ כל האמור הנ"ל, אנו מעוניינים בשדרוג ה-PSO הראשון בלבד, כלומר רק עבור פתרון RCPSP. מכיוון שאין שינוי במספר החלקיקים ו/או האיטרציות הרי שאין שינוי במספר הפעמים בהם אנו קוראים לפונקצית המטרה של RCPSP. המחיר של שימוש בMGPSO אם כן נובע בעיקר מפעולת מיון המיקומים המתבצעת בתחילת האלגוריתם (כחלק מהקצאת הטריטוריות). פעולת המיון מתבצעת פעם אחת בלבד במהלך כל הריצה, ואין היא מתארכת ככל שמספר הפעילויות בפרויקט גדל מכיוון שהיא תלויה במספר החלקיקים בלבד, שכאמור נשאר קבוע. משקיבלנו מספר פתרונות (במקום פתרון יחיד), הרי שיש לטפל בכולם. המשמעות היא שיש להפעיל את פתרון RDTP על כל אחד מארבעת התזמונים שהתקבלו מפתרון RCPSP. נזכור שבחרנו לממש את MGPSO עם ארבע קבוצות; המשמעות היא שהכפלנו פי 4 את זמן פתרון RDTP המצומצמת, כיוון שלמעשה אנו פותרים אותה ארבע פעמים! אך אל דאגה, הדבר פחות נורא ממה שהוא נשמע. נבין מדוע: כמו שכבר ציינו, מספר הפעילויות גדול הרבה יותר ממספר מועדי השחרור. מה עוד שאנו מניחים תמיד 4 מועדי שחרור, כלומר מספר מועדי השחרור אינו הולך וגדל ככל שישנן יותר פעילויות בפרויקט. לכן, מרבית זמן הריצה מושקע בפתרון של RCPSP ורק שבריר ממנו מושקע בפתרון של RDTP. על כן, הכפלה שלו ב-4 הינה מתקבלת על הדעת. אם כך, נוספו לנו 2 אלמנטים לזמן הריצה אך כל אחד מהם הינו קצר יחסית ובלתי תלוי במספר הפעילויות בפרויקט. זהו אם כן מחיר המתקבל על הדעת בתמורה להגדלה משמעותית בכושר האלגוריתם שלנו.

5.3. שימוש במבני נתונים יעילים

מבני נתונים יעילים הם מפתח לכתיבת אלגוריתמים יעילים. בהינתן נתונים רבים לא מעשי לתחזק עבור כל אחד מהם משתנה בודד משל עצמו. למשל: בהינתן משתנה המכיל זמן סיום של פעילות בפרויקט, הרי שאם רשת הפרויקט מכילה 120 פעילויות, היינו צריכים להגדיר 120 משתנים כאלו, וכן לכתוב עבור כל אחד מהם פונקציה המעדכנת אותו ואשר קוראת אותו. אם אנו מעוניינים גם בזמני ההתחלה של הפעילויות, אזי היינו צריכים לכתוב 240 פונקציות קריאה ועדכון, וזו רק ההתחלה! לעזרתנו נחלצים מבני הנתונים. הדוגמא שלעיל ממחישה היטב את סדר הגודל של הבעיה הניצבת בפנינו. אכן, בבואנו לשבץ פעילות חדשה על הגאנט אנו זקוקים למידע בדבר מועד תחילתן וסיומן של הפעילויות שכבר שובצו, ע"מ שנוכל לעמוד באילוצי הקדימויות והמשאבים. **המערך**, מבנה נתונים מוכר ופשוט, נותן מענה לבעיה זו. ניתן לשמור את כל זמני ההתחלה של הפעילויות במערך יחיד, ששמו נאמר Start_Times. ע"מ לגשת לזמן תחילתה של פעילות כלשהי i יש לגשת לכתובת Start_Times[i]. מרגע הקמת המערך, אין אנו זקוקים לכתוב פונקציות כתיבה וקריאה עבור כל פעילות בנפרד, אלא מספיק לכתוב אחת בלבד אשר תקבל כקלט את האינדקס i של הפעילות שבה אנו מעוניינים באותו הרגע. המערך הינו כלי חזק שכן הוא יוצר לנו סדר וארגון וזאת מבלי לאבד את הגישה האקראית לכל אחד מהאלמנטים שהוא מכיל. הצרות מתחילות כאשר אנו מעוניינים לגשת לפעילות דרך זמן ההתחלה (או הסיום) השלה ולא להיפך. בתיאוריה, ניתן ליצור מערך של נקודות זמן (נזכור שאנו מניחים שהזמן הינו בדיד ונמדד בימים שלמים). כל תא ייצג יום בפרויקט ויכיל את הפעילות שמתחילה באותו היום. אין אנו

יודעים מראש את משך הפרויקט, אך ניתן לשים לו גבול עליון ע"י למשל סכימה של משכי כל הפעילויות (הנחה שכל הפעילויות מתקיימות בטור). בשיטה זו נקבל מערך ארוך מאוד, קרוב לוודאי ארוך מדי, ובזבזני מבחינת תקורת הזיכרון. שיטה חמדנית פחות אולי תייצר מערך קצר יותר אך תסתכן בכך שהמערך יהיה קצר מדי, כלומר קצר מאורך הפרויקט בפועל. אף חמור מכך, ייתכן מצב אשר באותו היום מתחילה או מסתיימת יותר מפעילות אחת ועל כן תתרחש דריסה של נתונים. גם בעיה זו ניתן "לפתור", ע"י למשל קביעה ש-X התאים הראשונים מתייחסים ל-t=1, ה-X הבאים ל-t=2 וכך הלאה. שוב, פתרון בזבזני למדי.

אולם הבעיה החמורה מבחינתנו, היא לאו דווקא בעיית האחסון אלא בעיית הביצועים. נמחיש את הבעיה בעזרת הדוגמא הבאה. נניח:

- נניח פרויקט בעל n פעילויות.
- נניח שקיימים המערכים Start_Times ו-End_Times, אורכו של כל אחד מהם כמס' הפעילויות בפרויקט - n .
- נניח ששיבצנו חמש פעילויות כלשהן על הגאנט וכעת אנו מעוניינים לשבץ את הפעילות השישית, פעילות X.
- נניח גם שאנו יודעים שלפעילות X אין פעילויות קודמות.

המטרה היא לאתר את נקודת הזמן המוקדמת ביותר שבה פעילות X יכולה להתחיל מבלי להפר את אילוץ המשאבים. בכדי לבצע זאת, עלינו לסרוק את זמני ההתחלה והסיום של הפעילויות המשובצות ע"מ לקבוע את העומס על המשאבים בכל נקודת זמן. נשים לב שהמערכים שהגדרנו אינם מספקים לנו מידע ישיר לגבי אלו פעילויות כבר תוזמנו, על כן עלינו לסרוק את **כל** מערך ה-Start_Times בשביל למצוא אותן (לצורך העניין, פעילויות שטרם תוזמנו מתחילות בזמן t=(-1)). המצב לא היה שונה אילו בחרנו להשתמש במערך של נקודות זמן, עדיין היינו צריכים לסרוק את **כל** המערך ולהחליט עבור כל יום בנפרד אלו פעילויות מתחילות/מסתיימות בו. אם ניקח בחשבון שיש לנו n פעילויות בפרויקט, הרי שנשקיע בסופו של דבר n^2 פעולות לחישוב העומס על המשאבים (יש לחשב את העומס עבור כל פעילות בנפרד). משך פעולה בודדת כמובן אינו מידי ולוקח נאמר 1/1000 שנייה (לצורך המחשה). אם כך, הרי שעבור פרויקט בעל 100 פעילויות נזדקק ל $\frac{100^2}{10^3}=\frac{1}{10}$ שניות (=עשירית שנייה). וזאת מבלי להזכיר את שאר החישובים הכרוכים בבעיית התזמון. יש לזכור כי מכפלת האיטרציות במספר החלקיקים שקיימים באלגוריתם ה-PSO יכולה להגיע למאות! באלגוריתם ה-PSO שמימשנו מספר החלקיקים הוא 100 ואילו מספר האיטרציות הוא 30. אי לכך סך משך זמן הפתרון יכול לנוע סביב $\frac{1}{10}\cdot 100\cdot 10=100$ שניות שהן כ-1.7 דקות! נעיר כי הזמנים בדוגמא הם להמחשה בלבד אולם בפועל, התוכנית המקורית (שלפני שיפור הביצועים) רצה כ-2 דקות עבור פרויקט בעל 120 פעילויות. זהו זמן בלתי הגיוני בעליל; סביר כי המשתמש הממוצע יהפוך חסר סבלנות במקרה הטוב ובמקרה הרע אף יעזוב את עמדת העבודה עוד בטרם סיום ריצת התוכנית. בטח ובטח שזהו זמן בלתי הגיוני לצורך חקר האלגוריתם, שכן במסגרת

המחקר יש להריץ את האלגוריתם עשרות רבות של פעמים. כדי לשפר את הביצועים, החלטנו להשתמש במבנה הנתונים "רשימה מקושרת", אך כפי שנראה מיד, הרשימה המקושרת חילצה אותנו מצרה אחת והפילה אותנו בפח אחר. בסופו של דבר, פיתחנו מבנה נתונים ייחודי שהוא הכלאה בין "רשימה מקושרת" לבין מבנה נתונים הנקרא "טבלת גיבוב". למבנה הנתונים החדש אנו קוראים "רשימה מגובבת". להלן נביא תיאור קצר של השלושה:

5.3.1. שימוש ברשימה מקושרת

בהינתן מספר פעילויות בפרויקט יש צורך באותו מספר של רשומות, וע"מ להימנע משכפול של פונקציות תחזוקה, כפי שראינו, ניתן ליצור מערך של רשומות. אך אין בזה כדי לקדם אותנו מבחינת הבעיה שהוצגה לעיל, אנו עדיין נצטרך לסרוק את **כל** המערך כדי לחשב את העומס על המשאבים בנקודת זמן (ולכן נזנח מעתה את המערך בדיון על הרשימה המקושרת).

- **שימוש ברשימה מקושרת לייצוג רצף זמנים:**

נוח לייצג באמצעות הרשימה המקושרת רצפים (sequences) – אלמנטים המקושרים ביניהם בסדר שהוא בעל משמעות. באמצעות הגדרה של הרשומה הבאה ניתן לייצר רצף של זמני תחילת הפעילויות:

Link

Start_Time

Activity

Next_Link

עבור כל פעילות חדשה שאנו משבצים, נוכל להוסיף חוליה חדשה כנ"ל לרצף. לאחר שיבוץ של חמש הפעילויות הראשונות נקבל רשימה המכילה את חמשת זמני ההתחלה הראשונים. בהינתן כי סדר החוליות מבטא נכונה את סדר הזמנים, אין לנו צורך לסרוק את הרשימה כולה ע"מ למצוא את הפעילויות שמתחילות לפני נקודת זמן מסוימת, אלא יש לסרוק את הרשימה רק עד החוליה האחרונה המכילה את הזמן שבנדון. יש בכך כדי לצמצם בחצי(!) את כמות האיטרציות הנדרשות לחישוב העומס על המשאבים. נסביר מדוע: המקרה הגרוע הוא שהרשימה מלאה ויש לסרוק את כולה (n חוליות) והמקרה הטוב הוא שהרשימה ריקה (0 חוליות). המקרה הגרוע מתקבל כאשר אנו מעוניינים לשבץ את הפעילות האחרונה ובוחנים את העומס בנקודת הסיום של הפעילות *ששובצה* אחרונה. המקרה הטוב מתקבל כאשר אנו מעוניינים לשבץ את הפעילות הראשונה על הגאנט. אם כך, בממוצע נצטרך לסרוק n/2 חוליות עבור כל פעילות. נכפיל זאת ב-n פעילויות ונקבל $\frac{1}{2}n^2$ איטרציות, שיפור לא מבוטל (אם כי תלמידי מדמ"ח יגידו שהסיבוכיות נשארה זהה. בכל מקרה, מבחן התוצאה מראה כי חל שיפור בזמן הריצה).

- **אליה וקוץ בה**

ע"מ לקבל את שיפור הזמנים שתיארנו לעיל הנחנו כי אנו שומרים על סדר הזמנים בכל הוספה של חוליה חדשה. בפועל, זוהי משימה הצורכת זמן רב בפני עצמה. הוספה של חוליה במיקום **מסוים** היא כאמור פעולה פשוטה, הבעיה היא לקבוע **היכן** בדיוק המיקום המסוים שבו יש להוסיף חוליה חדשה. החיסרון של הרשימה המקושרת הוא שאנו לא יודעים מראש את הכתובת (בזיכרון) של כל חוליה אלא רק את הכתובת של החוליה הראשונה. כלומר, גישה אקראית נתונה רק עבור החוליה הראשונה, ומרגע שהגענו אליה אנו יכולים לשייט על פני הרשימה אך ורק באמצעות המצביעים שבתוך החוליות. ע"מ להוסיף חוליה חדשה, נאמר בעלת זמן התחלה Start_Time = 12, אנו נדרשים לסרוק את הרשימה מתחילתה עד אשר נפגוש בחוליה בעלת Start_Time > 12 או עד שנגיע לסוף הרשימה. כלומר נצטרך לסרוק בממוצע n/2 חוליות. בהתחשב בכל n הפעילויות, שמירה על הסדר התקין של רצף הזמנים תעלה לנו $\frac{1}{2}n^2$ איטרציות, כך שבסופו של דבר לא הרווחנו כלום. כדי להימנע מהתשלום על שמירת הסדר, ובמקביל להמשיך ליהנות מהאצת הביצועים שהרשימה המקושרת מספקת, ביצענו הכלאה בינה לבין מבנה הנתונים "טבלת גיבוב".

5.3.2. שימוש בטבלת גיבוב

היתרון של טבלת הגיבוב הוא שהיא מאפשרת לגשת לערכים שהיא מכילה במספר קטן מאוד של איטרציות. זאת בהינתן שפונקצית הגיבוב אכן מספקת תוצאות המתפלגות באופן אחיד וכן שהיחס בין מספר הערכים האפשריים לבין גודל הטבלה הוא מספיק נמוך. נסביר באמצעות הדוגמא הבאה: נאמר שבפרויקט מסוים ישנן 200 פעילויות ולכל אחת רשומה המכילה את שמה ואת זמן תחילתה. נאמר גם שלרשותנו טבלת גיבוב בגודל של 10 דליים. אם בידינו פונקצית גיבוב יעילה, הרי שבכל דלי יוקצו לכל היותר 20 פעילויות. תהליך החיפוש מתבצע כך: נניח שאנו מעוניינים לדעת את זמן תחילתה של פעילות מספר חמש. לשם כך עלינו לגשת לרשומה של פעילות זו. נכניס לפונקצית הגיבוב את הקלט 5 ונקבל כפלט את אינדקס התא (הדלי) במערך (בטבלת הגיבוב) שבו פעילות חמש נמצאת. כעת ניגש לרשימה המקושרת הנמצאת באותו התא ונסרוק אותה. הגישה לתא במערך היא גישה אקראית ואילו סריקת הרשימה אורכת לכל היותר מספר צעדים כמספר החוליות שבה. על כן במקרה הגרוע משך החיפוש בדוגמא לעיל לא יעלה על 21 פעולות. ובאופן כללי: בהינתן n פעילויות בפרויקט, וטבלת גיבוב בגודל m, משך החיפוש יהיה לכל היותר n/m (תחת הנחת פונק' גיבוב יעילה). אם נשמור את היחס n/m קבוע ומספיק קטן, הרי שמשך החיפוש יהיה קבוע ללא תלות ב-n (בניגוד לרשימה מקושרת שבה משך החיפוש הממוצע הוא n/2).

5.3.3. פיתוח מבנה נתונים יעיל ייחודי

אף על פי שטבלת הגיבוב מספקת ביצועים מעולים, ואע"פ שהיא משתמשת באופן טבעי ברשימות מקושרות, היא לא מספיק טובה לצרכים שלנו. ניזכר שאנו מעוניינים לסרוק **ברצף** את זמני ההתחלה של

הפעילויות ששיבצנו ע"מ לחשב את העומס בנקודת זמן. הסגמנטציה הטבעית של טבלת הגיבוב, הנובעת מהחלוקה לדליים, קוטעת לנו את הרצף בין החוליות ומקשה עלינו לסרוק אותן. לכל רשימה, שבתוך כל דלי, יש סוף רשימה והיא אינה מחוברת לרשימה הבאה. כדי להנות מכל העולמות יצרנו מבנה נתונים חדש לו אנו קוראים "רשימה מגובבת" (לא להתבלבל עם "רשימת גיבוב"). הרשימה המגובבת היא למעשה טבלת גיבוב בתוספת היכולת להפוך לרשימה מקושרת רגילה, ובחזרה. המהפך נעשה בעת מתן הפקודה המתאימה והוא מתבצע ע"י הפניה של כל מצביע של חוליה אחרונה בכל דלי אל עבר החוליה הראשונה של הרשימה הבאה בתור. נציין רק שיש להתגבר על מוקשים בדמות דליים ריקים ואובדן של מצביעים אשר יכול למנוע חזרה למצב הקודם. הרשימה המגובבת שיצרנו כמובן מתגברת על המוקשים הללו, אך חסרונה הוא שלא ניתן להוסיף או להחסיר חוליות כאשר היא נמצאת במצב של רשימה מקושרת, אלא יש לדאוג להחזיר אותה למצב של טבלת גיבוב. המעבר בין המצבים לוקח מספר צעדים כמספר הדליים בטבלת הגיבוב ועל כן הוא קבוע ואינו תלוי בכמות הערכים המאוחסנים ברשימה המגובבת (או בכמות הפעילויות). עבור הפרויקט שלנו בחרנו לשמור על יחס של 10 פעילויות (לכל היותר) לכל דלי. עבור פרויקט בעל 120 פעילויות נוכל להסתפק ב-12 דליים בלבד. באמצעות הרשימה המגובבת אנו יכולים לשמור על החיסכון של $\frac{1}{2}n^2$ איטרציות בעת חישוב העומס על המשאבים בנקודת זמן, בממוצע.

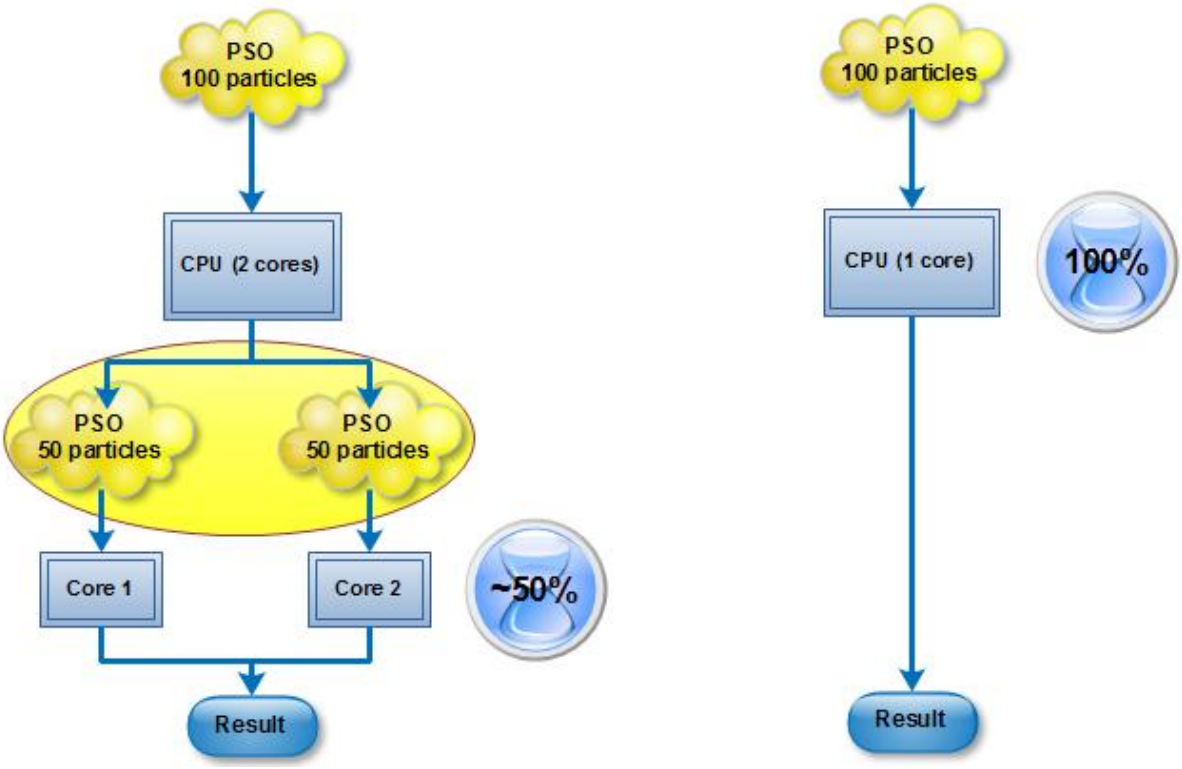
אם החיסכון של $\frac{1}{2}n^2$ איטרציות נראה כמו עבודה קשה עם שכר מועט בצידה, הרי שיש להכפיל אותו במספר החלקיקים שבאלגוריתם ה-PSO ואת המספר שמתקבל יש להכפיל במספר האיטרציות שאלגוריתם ה-PSO מבצע, וזאת ע"מ לקבל אומדן על המספר **המצטבר** של האיטרציות שחסכנו. באלגוריתם ה- PSO שמימשנו ישנם 100 חלקיקים ו-30 איטרציות. החיסכון המצטבר עבור פרויקט בעל 120 פעילויות הוא בסדר גודל של 10^7 איטרציות! לאחר שימוש ברשימה המגובבת הצלחנו לצמצם את זמן הריצה על פרויקט בעל 120 פעילויות משתי דקות לכ-57 שניות.

5.4. שימוש בבו זמניות (concurrency)

טכניקת הבו-זמניות מהווה את התותחים הכבדים בכל הנוגע לשיפור ביצועים. בו-זמניות אמיתית יכולה לשפר פלאים ביצועים, בעיקר עבור פתרון של בעיות הניתנות לפירוק לבעיות קטנות יותר. PSO הוא סוג כזה של בעיה, כיוון שניתן לחלק אותו לחלקים קטנים – כמספר החלקיקים. בפרויקט זה הרצנו את האלגוריתם על מחשב בעל מעבד כפול-ליבה (שתי ליבות) ולכן מימשנו אותו כך שיתפצל לשני חלקים: כל חלק יטפל בחצי מהחלקיקים. כמובן שבאיזושהי נקודה עליהם להיפגש יחד בכדי לספק פתרון יחיד. התענוג של האצת הביצועים באמצעות בו-זמניות אינו חינם, אלא הוא דורש מהתוכניתן לכתוב את התוכנית בצורה מאוד מסוימת ולדלג מעל מוקשים רבים כמו נעילות ודריסת נתונים הדדית, הנובעים מכך שכל החלקים של התוכנית בסופו של דבר מנסים לגשת לאותם משאבי זיכרון (לאותם משתנים). אין ספק

שהרווח הוא כולו של המשתמש; לאחר שימוש בבו-זמניות הצלחנו לצמצם את זמן הריצה על פרויקט בעל 120 פעילויות מ-57 שניות לכ-20 שניות!!

הערה: היה צפוי כי השיפור יתבטא במחצית הזמן ההתחלתי לכל היותר, אולם השיפור בפועל הוא גדול הרבה יותר. נסביר מדוע: הזכרנו כבר שגם לתוכנית בודדת יכולות להיות משימות רבות לעיבוד במקביל, אותו הדבר תקף גם לגבי המהדר (compiler) שבחרנו להשתמש בו לכתיבה והרצה של התוכנית שלנו. על כן, כאשר המהדר מקבל את תורו על המעבד, אין זה אומר בהכרח שהוא עסוק בלקדם את ריצת התוכנית שלנו אלא יתכן מאוד שהוא עסוק בלנהל את משאבי הזיכרון (למשל שחרור משתנים שכבר אין בהם שימוש). שימוש בטכניקת הבו-זמניות נותן לכל אחד משני החלקים מעמד של תוכניות עצמאיות, אשר נקראות גם Threads. אי לכך, הן מקבלות מקום משל עצמן בתור המשימות לעיבוד ממש כמו המהדר עצמו. הדבר מזכיר קבלה של כרטיס כניסה VIP בדלת האחורית בזמן שלפני הדלת הקדמית משתרך תור ארוך של אנשים "פשוטים".



איור 5.4.1 – הדגמת חלוקת עבודה על מעבד כפול ליבה.

על אף ששני החלקים של התוכנית פועלים על ליבות שונות, אין לראות בהן שתי ריצות שונות של PSO.

6. מהלך המחקר

6.1. ניסוח מטרת המחקר

מטרת המחקר הינה לבחון את טיב הכלי המוצע על ידינו לפתרון בעיית RDTP המורחבת, הן מבחינת זריזות אספקת הפתרונות והן מבחינת כושרו במציאת פתרונות עדיפים. המדידה תיעשה בשלושה מישורים:

א. ביצועים – יימדד משך זמן הריצה הממוצע של האלגוריתם מרגע הפעלתו ועד לאספקת תוצאה סופית על ידו: $\bar{t}$.

ב. כושר האלגוריתם – יבוטא באמצעות שני מדדים:

- **עוצמה** – לכאורה, נרצה למדוד את הענ"ן הממוצע המסופק ע"י האלגוריתם: $\overline{NPV}$. אך היות וההרצות אינן מתבצעות כולן על אותה רשת, והיות ולכל רשת פתרון אופטימאלי משלה, אין אנו יכולים להשתמש ב-NPV המתקבל כפי שהוא. על כן, יצרנו מדד הקרוי POM – Percentage of Maximum והוא מחושב ע"י חלוקה של הענ"ן המתקבל בריצה נתונה בענ"ן המקסימאלי הידוע לאותה הרשת. $POM_i = \frac{NPV_i}{NPV_{\max}}$. ההשוואה עצמה תיעשה על בסיס ה-POM הממוצע: $\overline{POM} = \frac{\sum POM_i}{n}$.
- **עקביות** – תימדד ע"פ סטיית התקן של ה-POM המתקבלים מהפתרונות המסופקים: $\sqrt{S^2_{POM}}$.

ג. ורסטיליות – שני המדדים הנ"ל יימדדו בנפרד על קבוצת פרויקטים בגודל בינוני (90 פעילויות) ועל קבוצת פרויקטים בגודל גדול (120 פעילויות). לאחר מכן נשווה בין יכולות האלגוריתם בין שתי הקבוצות בחיפוש אחר שוני הנובע מגודל הבעיה. ורסטיליות איתנה מבחינתנו היא כזו אשר לא תביא להבדלים גדולים בין הקבוצות מבחינת סטיית התקן (הנמדדת ב-POM).

6.2. אופן המדידה והסקת המסקנות

6.2.1. אלגוריתמי הבקרה

ע"מ להסיק מסקנות ראויות באשר ליכולותיו של הכלי המוצע, עלינו לבחון אותו אל מול אלגוריתמים נוספים אשר ישמשו כאלגוריתמי בקרה. נשתמש בארבעה אלגוריתמי בקרה: שני אלגוריתמים דטרמיניסטיים ושניים סטוכסטיים. מדד העקביות רלוונטי גם עבור האלגוריתמים הדטרמיניסטיים.

- <u>אלגוריתם דטרמיניסטי 1 – אלגוריתם חמדן – לפי גובה הערך המתקבל מפעילות</u>:

1. האלגוריתם משבץ קודם את הפעילות בעלת הענ"ן העצמי הגבוה ביותר, לאחר מכן את הפעילות עם הערך הגבוה ביותר הבא בתור וכו'. עבור פעילויות עם ערך זהה, קובע האינדקס של הפעילות: פעילות עם אינדקס נמוך יותר תשובץ קודם.

2. לאחר השלמת תזמון הפעילויות, האלגוריתם מתזמן את מועדי השחרור עם מרווח אחיד ביניהם ומחשב ענ"ן הפרויקט.

הרציונל: ככל שפעילות בעלת ערך מסתיימת מוקדם יותר, כך יש סיכוי רב יותר שתיכנס לגרסה מוקדמת והערך שהיא מפיקה יתקבל מוקדם יותר. ככל שרווח מתקבל מוקדם יותר, כך הוא נשחק פחות מהריבית בעת חישוב הענ"ן.

- <u>אלגוריתם דטרמיניסטי 2 – אלגוריתם חמדן – לפי משך הפעילות</u>:

1. האלגוריתם משבץ קודם את הפעילות בעלת המשך הקצר ביותר, לאחר מכן את הפעילות עם המשך הקצר ביותר הבא בתור וכו'. עבור פעילויות עם ערך זהה, קובע האינדקס של הפעילות: פעילות עם אינדקס נמוך יותר תשובץ קודם.
2. לאחר השלמת תזמון הפעילויות, האלגוריתם מתזמן את מועדי השחרור עם מרווח אחיד ביניהם ומחשב ענ"ן הפרויקט.

הרציונל: האלגוריתם מניח כי שיבוץ של פעילויות קצרות תחילה, יגרור משך פרויקט קצר יותר. משך פרויקט קצר יותר משמעותו שכל מועדי השחרור קרובים יותר לזמן תחילת הפרויקט, ולכן הערך המתקבל מהם צפוי להישחק פחות מהריבית בעת חישוב הענ"ן.

- <u>אלגוריתם סטוכסטי 1 – אלגוריתם אקראי– פשוט</u>:

בתחילה, היינו מעוניינים באלגוריתם אקראי פשוט *ושהוא* - עם מינימום של לוגיקה:

1. נבחרת נקודה יחידה במרחב הפתרונות של RCPSP ועל-פיה נקבע תזמון לפעילויות.
2. נבחרת נקודה יחידה במרחב הפתרונות של RDTP ועל-פיה נקבעים מועדי השחרור ומחושב ענ"ן הפרויקט.

בפועל, אלגוריתם זה הניב תוצאות עלובות למדי אשר נבעו מבחירה בלתי הגיונית* של מועדי שחרור אשר הביאה לעיתים קרובות מדי לענ"נים שליליים(!). ע"מ לספק לאלגוריתם יותר כוח תחרותי מול שאר האלגוריתמים (מה שבאנגלית מכונה fighting chance) הוספנו את הלוגיקה המינימליסטית הבאה הצבועה באדום:

1. נבחרת נקודה יחידה במרחב הפתרונות של RCPSP ועל-פיה נקבע תזמון לפעילויות.
2. נבחרת נקודה יחידה במרחב הפתרונות של RDTP ועל-פיה נקבעים מועדי השחרור ומחושב ענ"ן הפרויקט.
3. מחושב ענ"ן הפרויקט על סמך תזמון אפס** של מועדי השחרור.
4. מתבצעת השוואה בין הפתרונות המתקבלים מ-2 ו-3 ונבחרת התוצאה הגבוהה יותר.

** למשל: שחרור בימים עוקבים, או עם הפרש מועט ביניהם, בתחילת הפרויקט.*

*** תזמון אפס של RDTP, כפי שנובע מהגדרת המעטפת, הוא תזמון הקובע את מועד השחרור הראשון להיות באמצע הפרויקט, את מועד השחרור השני להיות במחצית של הזמן הנותר עד סוף הפרויקט וכך הלאה (אבל תמיד מועד השחרור האחרון הוא בסוף הפרויקט).*

- <u>אלגוריתם סטוכסטי 2 – אלגוריתם אקראי טהור – רחב</u>:

1. מחושבים מועדי שחרור משוערים על סמך תזמון אפס*.
2. נדגמות X נקודות במרחב הפתרונות של RCPSP, כך ש-X הוא מספר הקריאות לפונקצית המטרה של RCPSP ע"י אלגוריתם ה-PSO. X=30*100 = 3,000 .
3. נבחר התזמון המיטבי מבחינת ענ"ן עם מועדי השחרור המשוערים.
4. נדגמות Y נקודות במרחב הפתרונות של RDTP, כך ש-Y הוא מספר הקריאות לפונקצית המטרה של RDTP ע"י אלגוריתם ה-PSO. Y=30*100*4 = 12,000 .

**תזמון אפס של RCPSP: בכל צעד מתזמנים את הפעילות האמצעית ברשימת הפעילויות הפתוחות לתזמון.*

נדגיש כי האלגוריתם האקראי הרחב מהווה את אבן הבוחן המשמעותית ביותר מבחינתנו. הוא יריב ראוי בהחלט לאלגוריתם המוצע על ידינו בכך שהוא: **(א)**מנסה אלפי פתרונות, כמספר הפעמים שאלגוריתם ה-PSO קורא לפונקציות המטרה. זהו מספר רב מאוד של פעמים ועל כן צפוי שהוא יניב פתרונות איכותיים בממוצע. **(ב)**האלגוריתם האקראי הרחב הוא אלגוריתם אקראי *טהור*, משמע אין לו לוגיקה מסובכת כמו למשל הנעת חלקיקים במרחב, על כן הוא צפוי להיות מהיר יותר מאלגוריתם מבוסס ה-PSO. הקרב בין השניים צפוי להיות צמוד והציפייה היא שיוכרע במדד העקביות, לטובתו של האלגוריתם מבוסס ה-PSO.

הערה: כדי לפתח תחרות הוגנת, אלגוריתם ה-PSO שמימשנו לא משתמש באלגוריתם חמדן לקביעת אומדן משך הפרויקט, אלא הוא משתמש בתזמון האפס של RCPSP . כך, לאלגוריתמים החמדניים יש הזדמנות להוכיח את כוחם מול הלוגיקה הבסיסית של PSO, ולאלגוריתם ה-PSO יש הזדמנות להוכיח את כוחו ללא תלות באלגוריתמים החמדניים.

6.2.2. <u>מספר המדגמים ואופיים</u>

המדידות התבצעו על פני מספר רב של מדגמים; האלגוריתמים הורצו על 60 פרויקטים שונים – 30 פרויקטים בעלי 90 פעילויות ו-30 פרויקטים נוספים בעלי 120 פעילויות. על כל פרויקט הופעל כל אחד מהאלגוריתמים הסטוכסטיים 10 פעמים. סה"כ ריצות לאלגוריתם סטוכסטי: 600 ולאלגוריתם דטרמיניסטי: 60. רשתות הפעילויות נלקחו ממאגר PSPLIB – Project Scheduling Problem Library (מקור [11]), אשר נחשב למאגר רשתות מקובל באקדמיה לעריכת מבחנים מסוג זה. אחד

היתרונות של המאגר הוא הנגשת הרשתות בפני כל חוקר, כך שהניסויים יכולים להיות משוחזרים ע"י חוקרים אחרים ו/או להוות בסיס להשוואה של כלים המתיימרים להיות יעילים יותר. יתרון נוסף הוא הגדלת האובייקטיביות של המחקר עקב כך שהרשתות לא נוצרו על ידינו. הרשתות של PSPLIB מכילות את סדר הקדימויות, משך הפעילויות, צריכת המשאבים שלהן ואת כמות המשאבים ההתחלתית, אך הן אינן מכילות מידע באשר לתזרים המזומנים. המידע החסר הושלם בצורה אקראית. הריצות בוצעו כולן על אותו מחשב – HP Pavilion dv6000 עם זיכרון 1 GB RAM ומעבד כפול ליבה Intel Duo CPU 1.86 GHz (נתוני יצרן).

6.2.3. קביעת NPVmax כבסיס לחישוב ציוני ה-POM

בספריית PSPLIB ניתן למצוא עבור כל רשת פתרון אופטימאלי, או פתרון היוריסטי מיטבי שהושג לאחר מאות אם לא אלפי הרצות של חוקרים מסביב לעולם. אולם, מדובר בפתרונות אופטימאליים למציאת *משך הפרויקט* הקצר ביותר ולא למציאת הענ"ן הגבוה ביותר. נזכיר כי נתוני תזרים המזומנים לא קיימים בספריית PSPLIB והם הושלמו על ידינו באופן אקראי ולכן – אין בידינו מידע על הפתרונות האופטימאליים של הרשתות הנבחנות, אשר כאמור נחוץ לחישוב ציון ה-POM. ע"מ להתגבר על הבעיה, השתמשנו באלגוריתמים הנבחנים עצמם – קבענו כי $NPV_{\max}$ עבור רשת מסוימת הוא ה-NPV המכסימלי המתקבל מתוך כל ההרצות במחקר, על פני כל האלגוריתמים.

6.2.4. הסקת המסקנות

במהלך המחקר נבחן ארבע השערות אפס:

1. האלגוריתם המוצע על ידינו נותן תוצאות טובות יותר מאלגוריתמי הבקרה.
2. האלגוריתם המוצע על ידינו נותן תוצאות עקביות יותר מאשר כל אחד מהאלגוריתמים האקראיים.
3. האלגוריתם המוצע על ידינו אינו חורג ביותר מ-25% בזמן הריצה ביחס לאלגוריתם האקראי הרחב.
4. אין הבדל בביצועי האלגוריתם המוצע, או כושרו, בין ריצה על פרויקט בן 90 פעילויות לבין ריצה על פרויקט בן 120 פעילויות.

נאשש או נפריך את השערות 1,3 באמצעות מבחן t לשני מדגמים, ברמת מובהקות $\alpha = 0.05$.
נאשש או נפריך את השערות 2,4 באמצעות מבחן f לשני מדגמים (כולל ניתוח שונות - ANOVA) ברמת מובהקות $\alpha = 0.05$.

6.3. קבועים וסדרי גודל בשימוש:

- קבועים הנוגעים לאלגוריתם ה-PSO:
  - מספר חלקיקים: 100.
  - מספר איטרציות: 30.
  - סה"כ קריאות לפונקצית המטרה של RCPSP: 3,000.
  - סה"כ קריאות לפונקצית המטרה של RDTP: 12,000.
  - מרחב הפתרונות: בין 0 ל- 4π , בכל מימד.
  - משקולות בקביעת המהירות: $\omega = 0.2,\ \ \varphi_g = 1,\ \ \varphi_p = 0.3$
  - חישוב מקבילי: פיצול של עד 2 תתי-תוכניות במקביל.

- קבועים הנוגעים ל-MGPSO:
  - מספר קבוצות: 4.
  - מספר חלקיקים בקבוצה: 25.
  - רוחב טריטוריה: 20% ממרחב הפתרונות.

- קבועים וסדרי גודל הנוגעים לרשתות:
  - כמות פעילויות: 90 או 120.
  - משך פעילות: ימים בודדים עד עשרות בודדות.
  - מספר משאבים מתחדשים: 4.
  - היקף משאבים: עשרות בודדות של יחידות למשאב.
  - צריכה של משאב: יחידות בודדות עד עשרות בודדות למשאב.

- קבועים וסדרי גודל הנוגעים לתזרים המזומנים:
  - מספר מועדי שחרור: 4 (כולל סיום הפרויקט).
  - קנס לחריגה ממשך סטנדרטי של גרסה: 100 ₪ ליום.
  - שווי הכסף (ריבית) – 1% ליום.
  - סיכוי של פעילות להיות נושאת ערך – 20% (כל פעילות חמישית בממוצע).
  - השקעה נדרשת בפעילות נושאת ערך – סדר גודל של מאות ₪ .
  - גובה תקבול בודד של פעילות נושאת ערך – סדר גודל של מאות ₪ .
  - מספר התקבולים של פעילות נושאת ערך – בודדים עד עשרות בודדות.

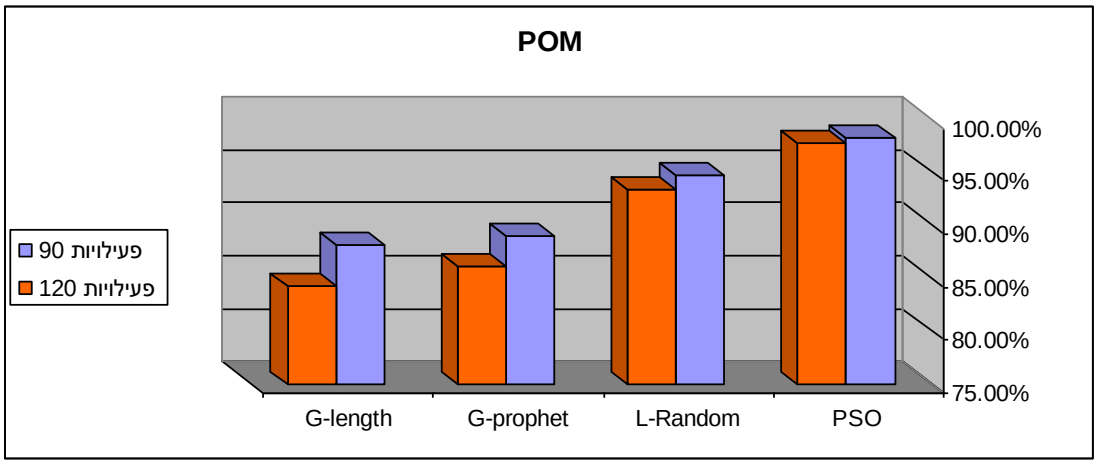
7. תוצאות המחקר

7.1. ערכי NPVmax

כאמור, הענ"ן המכסימלי עבור כל רשת נקבע כערך המכסימלי שהתקבל מאיזושהי הרצה של איזשהו אלגוריתם במהלך המחקר. לאחר כל ההרצות, התברר יתרון בולט לאלגוריתם ה-PSO . מתוך 60 רשתות – רק עבור ארבע מהן NPV_{max} *לא* נקבע באמצעות אלגוריתם ה-PSO. זהו כשלעצמו אומדן ליכולת הגבוהה של האלגוריתם. תוצאות המכסימום שהתקבלו מופיעים בנספח ג'.

7.2. השוואה בין ממוצע עוצמת הפתרונות, לפי מדד POM

כצפוי, אך לא מובטח מראש, אלגוריתם ה-PSO השיג את התוצאות הטובות ביותר, עם ממוצע של 97.96 [POM]. בצמוד לו, ניצב במקום השני האלגוריתם האקראי הרחב. האלגוריתם האקראי הפשוט מדשדש מאחור עם ממוצע של 47.39 [POM], כראוי לטבעו. מפתיעים ביכולתם אלו האלגוריתמים החמדניים אשר השיגו ציונים גבוהים במיוחד! מבחן t שנערך בין תוצאות ה-PSO לבין תוצאות האלגוריתם האקראי הרחב מראות על פער ממוצע של כ-4 [POM] אשר מובהק מבחינה סטטיסטית.



איור 7.2.1 – גרף השוואת ממוצעי הפתרונות, לפי מדד POM . לגרף הכולל את S-Random ראה נספח ב'.

	90 פעילויות	120 פעילויות	סה"כ
PSO	98.23%	97.70%	97.96%
L-Random	94.61%	93.34%	93.98%
G-prophet	88.90%	86.02%	87.46%
G-length	88.08%	84.16%	86.12%
S-Random	52.95%	41.84%	47.39%

טבלה 7.2.1 – ממוצעי הפתרונות, לפי מדד POM .

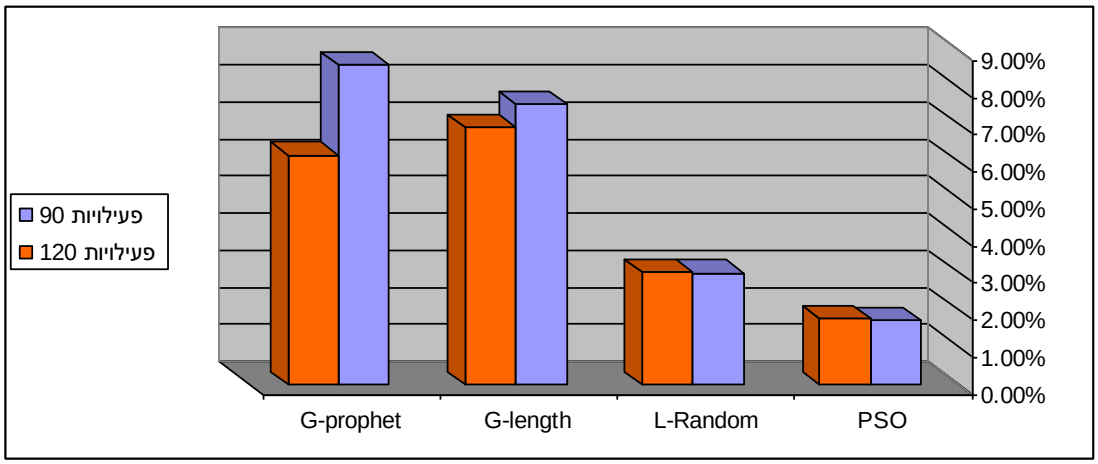
t-Test: Paired Two Sample for Means		
L-Random	***PSO***	
93.98%	97.96%	Mean
0.000951	0.000314399	Variance
600	600	Observations
	0.334750785	Pearson Correlation
	0	Hypothesized Mean Difference
	599	df
	32.55187993	t Stat
	6.818E-135	P(T<=t) one-tail
	1.647401445	t Critical one-tail
	1.3636E-134	P(T<=t) two-tail
	1.963932159	t Critical two-tail

טבלה 7.2.2 – מבחן t למציאת הבדל בין ממוצע הפתרונות של PSO לזה של האלגוריתם האקראי הרחב.

ההבדל נמצא מובהק.

7.3. השוואה בין עקביות הפתרונות, ע"פ סטיית התקן

עקביות הפתרונות הינה תכונה רצויה מאוד עבור אלגוריתם. בלעדיה, גם בהינתן שהאלגוריתם בעל ממוצע פתרונות גבוה, לא נוכל להיות בטוחים בטיב הפתרון המתקבל עבור ריצה מסוימת. במדד זה נחשפת חולשתם של האלגוריתמים החמדניים, אשר היו בעלי סטיית תקן שנעה סביב 7.5 [POM] לכל אחד. אלגוריתם ה-PSO הציג עקביות יוצאת דופן עם סטיית תקן ממוצעת של 1.77 [POM] בלבד! האלגוריתם האקראי הרחב הציג סטיית תקן של כ-3 [POM]. ההפרש בין סטיות התקן של אלגוריתם ה-PSO ושל האלגוריתם האקראי הרחב נמצא מובהק מבחינה סטטיסטית.



איור 7.3.1 – גרף השוואת סטיות התקן, לפי מדד POM . לגרף הכולל את S-Random ראה נספח ב'.

סה"כ	120 פעילויות	90 פעילויות	
1.77%	1.78%	1.73%	**PSO**
3.08%	3.04%	3.01%	**L-Random**
7.47%	6.95%	7.57%	**G-length**
7.58%	6.18%	8.62%	**G-prophet**
17.44%	12.24%	19.94%	**S-Random**

טבלה 7.3.1 – סטיות התקן, לפי מדד POM.

F-Test Two-Sample for Variances		
PSO	***L-Random***	
0.979625	0.939766	Mean
0.000314	0.000951	Variance
600	600	Observations
599	599	df
	3.025952	F
	3.37E-40	P(F<=f) one-tail
	1.143988	F Critical one-tail

טבלה 7.3.2 – מבחן f (פשוט) למציאת הבדל בשונות של פתרונות ה- PSO מול האלגוריתם האקראי הרחב.

ההבדל נמצא מובהק.

ס. תקן	ממוצע	
1.77%	97.96%	**PSO**
3.08%	93.98%	**L-Random**
7.47%	87.46%	**G-length**
7.58%	86.12%	**G-prophet**
17.44%	47.39%	**S-Random**

טבלה 7.3.3 – טבלה מסכמת, ס. תקן וממוצע (סה"כ). אלגוריתם ה-PSO נמצא עדיף בשניהם.

7.4. מדידת שונות פנימית באלגוריתם ה-PSO

ביצענו מבחן f (ניתוח שונות) למציאת הבדל בשונות הפתרונות של אלגוריתם ה-PSO, בין פרויקטים בעלי 90 פעילויות לבין פרויקטים בעלי 120 פעילויות, כמדד לורסטיליות. נמצא הבדל זעיר של 0.05 [POM] בין סטיות התקן של הפתרונות בשתי קבוצות הפרויקטים, אך מובהק מבחינה סטטיסטית.

לסקירת תרשימי היסטוגרמה של כלל האלגוריתמים בחלוקה לפי כמות פעילויות ראה נספח ד'.

Anova: Single Factor				
SUMMARY				
Variance	*Average*	*Sum*	*Count*	*Groups*
0.000299	0.982289	294.6868	300	**90-projects**
0.000316	0.976961	293.0883	300	**120-projects**

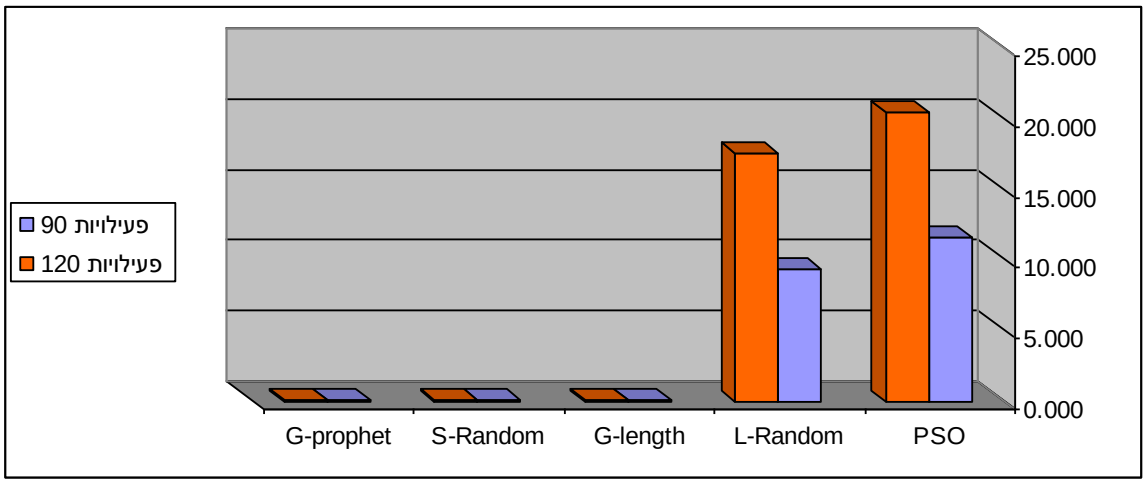
ANOVA						
F crit	*P-value*	*F*	*MS*	*df*	*SS*	*Source of Variation*
3.857056	0.000218	13.83611	0.004259	1	0.004258796	Between Groups
			0.000308	598	0.184066137	Within Groups
				599	0.188324933	Total

טבלה 7.4.1 – מבחן f למציאת הבדל בשונות של פתרונות ה- PSO בין קבוצות הפרויקטים. ההבדל נמצא מובהק.

7.5. השוואה בין זמני הריצה הממוצעים (בשניות)

זמני הריצה של האלגוריתמים החמדניים ושל האלגוריתם האקראי הפשוט הם זניחים לחלוטין, כיוון שהם קוראים לפונקציות המטרה מספר מצומצם מאוד של פעמים. התחרות היא בין אלגוריתם ה-PSO לבין האלגוריתם האקראי הרחב. שניהם קוראים לפונקציות המטרה אותו מספר של פעמים. במדד זמן הריצה מפסיד PSO לאלגוריתם האקראי הרחב עם תוספת זמן ריצה ממוצעת של כ-16.6% עבור פרויקטים עם 120 פעילויות, ותוספת של כ-24.4% עבור פרויקטים עם 90 פעילויות.

נעיר כי גם האלגוריתם האקראי הרחב נכתב כך שיעשה שימוש בכל האמצעים לשיפור הביצועים שעושה בהם שימוש אלגוריתם ה-PSO.



איור 7.5.1 – גרף השוואת זמני הריצה (בשניות). האלגוריתמים הפשוטים מנצחים בנוק-אאוט.

120 פעילויות	90 פעילויות	
20.419	11.592	**PSO**
17.512	9.312	**L-Random**
0.041	0.036	**G-length**
0.038	0.028	**S-Random**
0.028	0.019	**G-prophet**

טבלה 7.5.1 – זמני הריצה הממוצעים, בשניות.

8. מסקנות ודיון

- **כושר האלגוריתם במציאת פתרונות עדיפים:**

האלגוריתם המוצע על ידינו נותן תוצאות טובות יותר מאלגוריתמי הבקרה, בממוצע. מפתיעים בכושרם האלגוריתמים החמדניים וניכר כי הפערים בין האלגוריתמים השונים הם קטנים יחסית ונעים בין אחוזים בודדים לעד כעשרה אחוזים מהענ"ן המכסימאלי (להוציא האלגוריתם האקראי הפשוט). משמעות הדבר היא שההבדל בין האלגוריתמים נהיה משמעותי רק עבור פרויקטים בעלי תזרימי מזומנים גבוהים ביותר, שבהם לכל אחוז רווח/הפסד יש משקל כבד מבחינה פיננסית. עולה השאלה מדוע האלגוריתמים החמדניים הצליחו כל כך במדד ממוצע ציוני ה-POM . להלן נעלה מספר **השערות**:

א. האלגוריתמים החמדניים נותנים למעשה תוצאות בינוניות אך כך גם אלגוריתם ה-PSO ולכן הם מקבלים ציון גבוה במדד ה-POM שהינו מדד יחסי.

ב. שני האלגוריתמים מניחים ריווח אחיד בין מועדי השחרור. ריווח שכזה מצמצם לאפס את הקנסות. כמו כן, הוא מונע רווחים גדולים בין שחרור לשחרור מה שעוזר לצמצם את השחיקה של הרווחים הנובעת מהריבית. שני גורמים אלו מקרבים לאידיאל את הענ"ן, תוך השפעה מועטה של תזמון המוצא של RDTP (הפתרון של RCPSP).

ג. אילוצי הקדימויות נוקשים מדי וגורמים לכך שתזמון המוצא של RDTP דומה מדי בין ריצה לריצה, מה שמוביל לתוצאות דומות עבור כל אלגוריתם השואף לרווח באופן אחיד את מועדי השחרור.

נתייחס קודם להשערה א'. בהשערה זו עולה טענה כנגד כושרו של אלגוריתם ה-PSO שמימשנו. ע"מ לוודא כי כושר אלגוריתם ה-PSO מניח את הדעת הפעלנו אותו על מדגם של עשר רשתות של 120 פעילויות, אך במטרה למצוא את *משך הפרויקט המינימאלי*. משך הפרויקט המינימאלי הינו נתון שניתן להשוותו לערך הטוב ביותר שנמצא בספריית PSPLIB, וכך לקבל מדד על כושר האלגוריתם. להלן התוצאות:

רשת	Algoritm	Run No.	min found	known min	POM
j1203_1	PSO	1	89.00	80	89.89%
j1203_2	PSO	1	88.00	88	100.00%
j1203_3	PSO	1	102.00	100	98.04%
j1203_4	PSO	1	83.00	71	85.54%
j1203_5	PSO	1	92.00	84	91.30%
j1203_6	PSO	1	102.00	102	100.00%
j1203_7	PSO	1	96.00	93	96.88%
j1203_8	PSO	1	86.00	77	89.53%
j1203_9	PSO	1	86.00	86	100.00%
j1203_10	PSO	1	103.00	103	100.00%

	std dev	average
POM	**5.49%**	**95.12%**

טבלה 8.1 – תוצאות הרצת PSO למציאת המשך הקצר ביותר.

גם במקרה הזה השתמשנו במדד POM לצורך מתן הציונים. הפעם חישבנו את POM בצורה הבאה:

$POM_i = \frac{length_{\min}}{length_i}$, כאשר $length_{\min}$ נלקח מספריית PSPLIB . התוצאות מראות כי האלגוריתם הוא בעל כושר סביר, אם כי לא מושלם ועל כן יש להסיק כי התוצאות שקיבלנו במקרה של מציאת הענ"ן המכסימלי הן לא התוצאות האופטימאליות (שלא באופן מפתיע). אם כך, יש בכך כדי להסביר לפחות חלק, אך לא את רוב, ההצלחה של האלגוריתמים החמדניים במדד ה-POM. שיפור יכולותיו של אלגוריתם ה-PSO יכול להוות נושא למחקר המשך.

על ההשערות המנוסחות בסעיפים ב' + ג' ניתן לנסות לענות ע"י שינוי של האלגוריתם האקראי הפשוט, כך שבמקום לפתור את RDTP באמצעות תזמון אקראי (ו/או תזמון אפס), נפתור אותו תמיד עם ריווח אחיד של מועדי השחרור. פתרון ה- RCPSP נשאר אקראי. כדי לבחון את השיפור שחל, ביצענו הרצות מחודשות של האלגוריתם האקראי הפשוט על אחת הרשתות של 120 פעילויות:

		Original Simple Random					
		POM-NPV	Max NPV	NPV	Run No.	Algoritm	רשת
Std. dev	Average	42.09%	9,462.93	3,982.52	1	S-Random	j1203_10
5.03%	38.64%	39.01%	9,462.93	3,691.84	2	S-Random	
		41.86%	9,462.93	3,961.19	3	S-Random	
		36.21%	9,462.93	3,426.21	4	S-Random	
		39.74%	9,462.93	3,760.10	5	S-Random	
		48.22%	9,462.93	4,562.80	6	S-Random	
		37.75%	9,462.93	3,572.08	7	S-Random	
		33.58%	9,462.93	3,177.89	8	S-Random	
		38.29%	9,462.93	3,623.21	9	S-Random	
		29.69%	9,462.93	2,809.59	10	S-Random	

		Modified Simple Random					
		POM-NPV	Max NPV	NPV	Run No.	Algoritm	רשת
Std. dev	Average	92.07%	9,462.93	8,712.96	1	S-Random	j1203_10
2.59%	91.47%	92.82%	9,462.93	8,783.79	2	S-Random	
		91.22%	9,462.93	8,631.99	3	S-Random	
		90.09%	9,462.93	8,525.45	4	S-Random	
		94.70%	9,462.93	8,961.46	5	S-Random	
		90.02%	9,462.93	8,518.49	6	S-Random	
		88.88%	9,462.93	8,411.02	7	S-Random	
		89.10%	9,462.93	8,431.91	8	S-Random	
		89.16%	9,462.93	8,436.94	9	S-Random	
		96.58%	9,462.93	9,139.75	10	S-Random	

טבלה 8.2 – בחינת ההשפעה של ריווח אחיד של מועדי השחרור על ענ"ן הפרויקט.

למעלה: תוצאות האלגוריתם המקורי. למטה: תוצאות האלגוריתם לאחר השינוי.

התוצאות מראות שיפור משמעותי ביותר לאחר השינוי! המסקנה: ריווח אחיד של מועדי השחרור בהחלט מקפיץ את ענ"ן הפרויקט. היות ותזמון המוצא של RDTP נשאר אקראי, נשאלת השאלה: האם אכן לתזמון המוצא השפעה מועטה, או שמא אילוצי הקדימויות גורמים לכך שכל תזמוני המוצא האפשריים דומים מדי, תחת ריווח אחיד של מועדי השחרור?

כדי לנסות לענות על שאלה זו, לקחנו את אותה הרשת שהרצנו עליה את האלגוריתם האקראי הפשוט *המשופר* ומחקנו את כל אילוצי הקדימויות של 60 הפעילויות *הראשונות*. לאחר מכן הרצנו האלגוריתם מחדש וקיבלנו התוצאות הבאות:

		Modified Simple Random + modified j1203_10					
		POM-NPV	Max NPV	NPV	Run No.	Algoritm	רשת
Std. dev	Average	91.63%	10,984.00	10,065.14	1	S-Random	j1203_10
4.03%	88.01%	90.65%	10,984.00	9,956.53	2	S-Random	
		85.49%	10,984.00	9,390.47	3	S-Random	
		83.71%	10,984.00	9,194.90	4	S-Random	
		93.92%	10,984.00	10,316.47	5	S-Random	
		84.26%	10,984.00	9,255.22	6	S-Random	
		84.40%	10,984.00	9,270.10	7	S-Random	
		84.40%	10,984.00	9,270.10	8	S-Random	
		93.12%	10,984.00	10,228.04	9	S-Random	
		88.54%	10,984.00	9,724.93	10	S-Random	

טבלה 8.3 – בחינת ההשפעה של צמצום אילוצי קדימויות + ריווח אחיד של מועדי השחרור על ענ"ן הפרויקט.

בולטת לעין העובדה שהסרת האילוצים אפשרה קיומו של תזמון טוב יותר מהענ"ן המכסימאלי שנמצא מלכתחילה. בעמודת Max NPV נמצא הערך החדש לרשת המעודכנת לאחר הפעלה של אלגוריתם ה-PSO. ניתן לראות כי הסרת האילוצים פגמה ביכולת של האלגוריתם האקראי הפשוט המשופר, הן מבחינת ממוצע הפתרונות שלו והן מבחינת העקביות שלו. האם אלגוריתם ה-PSO ספג פגיעה דומה?

		PSO + modified j1203_10					
		POM-NPV	Max NPV	NPV	Run No.	Algoritm	רשת
Std. dev	Average	98.72%	10,984.00	10,843.00	1	PSO	j1203_10
0.99%	98.43%	100.00%	10,984.00	10,984.00	2	PSO	
		98.11%	10,984.00	10,776.71	3	PSO	
		98.79%	10,984.00	10,850.84	4	PSO	
		97.90%	10,984.00	10,753.18	5	PSO	
		99.94%	10,984.00	10,976.86	6	PSO	
		97.76%	10,984.00	10,737.65	7	PSO	
		97.37%	10,984.00	10,695.58	8	PSO	
		98.67%	10,984.00	10,838.10	9	PSO	
		97.08%	10,984.00	10,663.38	10	PSO	

טבלה 8.4 – בחינת ההשפעה של צמצום אילוצי קדימויות על אלגוריתם ה-PSO.

ע"פ המדגם הנ"ל של הרצות PSO על אותה הרשת המעודכנת התשובה היא לא. המסקנות אם כך הן:

א. הריווח האחיד של מועדי השחרור מביא להקטנת השונות בין ממוצעי פתרונות אלגוריתמי הבקרה לבין פתרונות אלגוריתם ה-PSO.

ב. ההשפעה הנ"ל קטנה ככל שאילוצי הקדימויות פחות נוקשים.

במבט לאחור אין אנו אמורים להיות מופתעים ממסקנה א' הנ"ל. הרי עצם קיומו של קנס לכל יום חריגה ממשך סטנדרטי לגרסה מפעיל לחץ על הפתרון האופטימאלי להיות בעל מרווח אחיד בין מועדי השחרור. הקנס שנקבע הינו 100 ₪ ליום חריגה והוא באותו סדר גודל כמו גובה התקבולים של הפעילויות נושאות הערך. נשאלת השאלה הם קבענו קנס גבוה מדי אשר הפעיל לחץ חזק מדי? כדי לענות על השאלה הזו, הפעלנו את אלגוריתם ה-PSO והאלגוריתם האקראי הפשוט המשופר *שוב* אך הפעם צמצמנו את קנס החריגה לכדי 10 ₪ ליום בלבד (הריצה היא עדיין על הרשת המעודכנת מבחינת אילוצי הקדימויות):

		Modified Simple Random + modified j1203_10 + low penalty					
		POM-NPV	Max NPV	NPV	Run No.	Algoritm	רשת
Std. dev	Average	82.54%	11,567.80	9,547.54	1	S-Random	j1203_10
2.94%	84.57%	86.03%	11,567.80	9,951.23	2	S-Random	
		85.57%	11,567.80	9,898.18	3	S-Random	
		83.50%	11,567.80	9,658.70	4	S-Random	
		83.26%	11,567.80	9,630.96	5	S-Random	
		85.95%	11,567.80	9,942.57	6	S-Random	
		85.83%	11,567.80	9,928.63	7	S-Random	
		88.56%	11,567.80	10,244.78	8	S-Random	
		86.56%	11,567.80	10,013.33	9	S-Random	
		77.92%	11,567.80	9,013.18	10	S-Random	

		PSO + modified j1203_10 + low penalty					
		POM-NPV	Max NPV	NPV	Run No.	Algoritm	רשת
Std. dev	Average	98.49%	11,567.80	11,393.42	1	PSO	j1203_10
1.12%	98.13%	97.51%	11,567.80	11,280.03	2	PSO	
		97.22%	11,567.80	11,246.59	3	PSO	
		99.99%	11,567.80	11,566.90	4	PSO	
		97.29%	11,567.80	11,253.93	5	PSO	
		97.01%	11,567.80	11,221.82	6	PSO	
		97.97%	11,567.80	11,333.03	7	PSO	
		97.27%	11,567.80	11,251.69	8	PSO	
		98.55%	11,567.80	11,399.78	9	PSO	
		100.00%	11,567.80	11,567.80	10	PSO	

טבלה 8.5 – בחינת ההשפעה של צמצום אילוצי קדימויות + הקטנת הקנס על חריגה ממשך גרסה סטנדרטי.

עם הקטנת גובה הקנס חלה פגיעה בכושר של שני האלגוריתמים, אולם הפגיעה באלגוריתם ה-PSO היא הרבה יותר קטנה מאשר הפגיעה באלגוריתם האקראי הפשוט המשופר. נציין שמועדי השחרור שהופקו ע"י אלגוריתם ה-PSO אינם בלתי הגיוניים כפי שאולי ניתן היה לצפות. יש היגיון מתמטי בריווח של מועדי השחרור גם ללא קנס חריגה, מאותן הסיבות שנמנו בתחילת הפרק. נסכם מסקנה סופית:

ככל שפרויקט מתאפיין במאפיינים הבאים כך הרווח המתקבל כתוצאה משימוש באלגוריתם ה-PSO גבוה יותר:

א. בעל היקף תזרים מזומנים גבוה.

ב. בעל מעט אילוצי קדימויות*.

ג. בעל גמישות גבוהה במשך הגרסאות.

**נדגיש כי אלגוריתם ה-PSO הוכח כעדיף גם עבור מקרים עם אילוצים רבים.*

- **עקביות התוצאות:**

אלגוריתם ה-PSO אינו מציג תוצאות יפות בממוצע בלבד, אלא הוא עקבי מאוד באספקת הפתרונות העדיפים. זאת בניגוד לאלגוריתמי הבקרה הפשוטים שהם בעלי סטיות תקן גבוהות יחסית. אין זה טריוויאלי שאלגוריתם סטוכסטי מספק פתרונות עם שונות כל כך נמוכה, ואנו זוקפים זאת בעיקר לזכות הטכניקה של MGPSO המאפשרת לברור בין מספר פתרונות. האלגוריתם האקראי הרחב מציג עקביות טובה מאוד גם כן, אך פחותה מזו של ה-PSO באופן מובהק.

- **זמן ריצה – יש תמורה בעד האגרה:**

מבחינת זמן הריצה, האלגוריתמים החמדניים הם המנצחים הגדולים; הם מספקים תוצאות יפות מאוד בכמעט אפס זמן. אלגוריתם ה-PSO והאלגוריתם האקראי הרחב מגרדים אחוזים נוספים תמורת זמן עיבוד נוסף, אך הזניח מבחינה יומיומית (פחות מחצי דקה). כמו כן, כפי שניתן לראות בפרק התוצאות, זמן הריצה של אלגוריתם ה-PSO אינו חורג ביותר מ-25% בזמן הריצה ביחס לאלגוריתם האקראי הרחב. בהינתן פרויקט עם תזרים מזומנים בנפח גבוה במיוחד, אין ספק ששווה להשקיע את זמן העיבוד הנוסף הזה בכדי להשיג שולי רווח גבוהים יותר.

- **ורסטיליות בין רשתות:**

עבור כל האלגוריתמים שהשתתפו במחקר קיים הבדל מובהק בין ממוצע (וסטיית התקן) של הפתרונות עבור פרויקט בן 90 פעילויות לבין ממוצע (וסטיית התקן) של הפתרונות עבור פרויקט בן 120 פעילויות (מבחינת ציוני POM). זוהי מסקנה שאינה מפתיעה, במיוחד לאור העובדה שבעיית RDTP הופכת להיות מורכבת יותר ככל שישנן פעילויות רבות יותר. על כן צפוי היה שטיב הפתרונות ידרדר עם הגידול במספר הפעילויות. אומנם האפקט היה נמוך מאוד עבור כל האלגוריתמים שנבדקו פרט לאלגוריתם האקראי הפשוט, אך הוא היה קטן במיוחד עבור אלגוריתם ה-PSO. קיימת פגיעה קטנה מאוד בכושר של PSO במעבר מרשת של 90 פעילויות לרשת של 120 ולכן, מבחינה זו, אלגוריתם ה-PSO הוא בעל מידה גבוהה מאוד של ורסטיליות.

9. סיכום

בתחילת הפרויקט, כשסקרנו את המניעים שלנו לעסוק דווקא בבעיית תזמון מועדי השחרור, ציינו את היעדרותה הכמעט מוחלטת של בעיה זו מן הספרות. אחת ההשערות שהעלנו להיעדרות הבולטת היא הקושי החישובי שכרוך בפתרון הבעיה. איפכא מסתברא – המחקר מצביע על כך שטכניקה פשוטה ביותר: ריווח אחיד בין מועדי השחרור, מביאה בחלק רב של המקרים לתוצאות יפות מבחינת השאת ענ"ן הפרויקט. ואולם, הטכניקה הנ"ל אינה מושלמת וכמעט תמיד אין היא מביאה לתוצאות מושלמות אלא לתוצאות הניתנות לשיפור. ואכן בליבה של הנדסת התעשייה והניהול לשפר ולייעל. וכאשר מעוניינים לשפר את הטוב למצוין, או אז מרים ראשו הקושי החישובי – ועומדת בפני מנהל הפרויקט השאלה האם ניתן להתיר את אותו הקושי. במחקר זה הראינו כי התשובה היא בהחלט כן. בפרויקט זה הראינו שניתן ליצור כלי מטה-היוריסטי אשר מביא לשיפור הענ"ן של הפרויקט, באופן עקבי, בזריזות רבה מאוד וקלה לשימוש. גם מנהלי פרויקטים בעלי תזרים מזומנים צנוע ודאי ישמחו להגדיל את שולי הרווח שלהם, על אחת כמה וכמה מנהלי פרויקטים בעלי תזרימי מזומנים של עשרות מיליוני דולרים אשר כל אחוז רווח הוא בעל משקל כבד ביותר. קביעה זו נכונה פי כמה כלפי פרויקטים המתאפיינים במיעוט אילוצי קדימויות ו/או בגמישות רבה יותר במשך הגרסאות. עבור פרויקטים שכאלה, ההתחבטות על סדר הפעילויות ועל מועדי השחרור היא גבוהה יותר ומשפיעה יותר על ענ"ן הפרויקט. כלי מטה-היוריסטי יכול לעזור לפתור את ההתחבטות במהירות וביעילות.

אנו בטוחים כי ניתן לשפר את הכלי שפותח במסגרת פרויקט זה, הן מבחינת כושרו במציאת פתרונות עדיפים והן מבחינת זמן הריצה. שיפור הכלי יכול להוות נושא למחקר המשך. כמו כן, אנו לא בטוחים מה היו התוצאות אילו ביצענו את המדידות על רשתות גדולות יותר (150 פעילויות ומעלה); האם הפער לטובתו של אלגוריתם ה-PSO היה בולט יותר, או היה נשאר דומה? גם שאלה זו יכולה להוות קרקע למחקר עתידי. אך יותר מכל, מחקר המשך מתבקש מבחינתנו הוא זיהוי ומיפוי של סוגי פרויקטים אמיתיים היכולים להנות מהכלי שפיתחנו; פרויקטים המתאפיינים עם מיעוט אילוצי קדימויות, היקף תזרים מזומנים גבוה וגמישות בקביעת מועדי השחרור.

10. ביבליוגרפיה

הערה: מקורות האיורים מצוינים בסמוך לאיורים עצמם והם אינם בהכרח מקורות המידע.

איורים חסרי ציון מקור הינם איורים מקוריים.

[1] Changsheng Zhang, Jiaxu Ning, Dantong Ouyang.
"A hybrid alternate two phases particle swarm optimization algorithm for flow shop scheduling problem", *Computers & Industrial Engineering, Vol. 58, No. 1 (2010).*
College of Information Science & Engineering, Northeastern University, Shenyang.

[2] D. Y. Shaa and Hsing-Hung Linb
"A multi-objective PSO for job-shop scheduling problems", *Expert Systems with Applications, Vol. 37, No. 2 (2010).*
Department of Industrial Engineering and System Management, Chung Hua University, Hsin Chu, Taiwan.

[3] Yuval Cohen, Arik Sadeh, Ofer Zwikael
"An Efficient Technique for Finding the Shortest Non-Delay Schedule for a Resource-Constrained Project", *Eur. Journal of Operational Research, Vol. 189, No. 3 (2008).*
Management & Economics Dept., the Open University of Israel, Israel.

[4] Kum Khiong Yang, F. Brian Talbot, James H. Patterson
"Scheduling a project to maximize its net present value: An integer programming approach", *Eur. Journal of Operational Research, Vol. 64, No. 2 (1993).*
Decision Sciences Department, National University of Singapore, Singapore.

[5] Robert A. Russel
"A comparison of heuristics for scheduling projects with cash flows and resource restrictions", *Management Science, Vol. 32, No.10 (1986).*
College of Business Administration, The University of Tulsa, Tulsa.

[6] D.C. Paraskevopoulos, C.D. Tarantilis, G. Ioannou
"Solving project scheduling problems with resource constraints via an event list-based evolutionary algorithm", *Expert Systems with Applications, Vol. 39, No. 4 (2012).*
Management Science Laboratory, Department of Management Science & Technology, Athens University of Economics & Business, Greece

[7] Ruey-Maw Chen a, Chung-Lun Wub, Chuin-Mu Wang a, Shih-Tang Lo
"Using novel particle swarm optimization scheme to solve resource-constrained scheduling problem in PSPLIB", *Expert Systems with Applications, Vol. 37, No. 3 (2010).*
Department of Computer Science and Information Engineering, National Chin-yi University of Technology, Taiwan.

[8] Mohammad Mahdi Nasiri
"A pseudo particle swarm optimization for the RCPSP", *Springer-Verlag London Limited (2012).*
Int J Adv Manuf Technol.

[9] Jang-Ho Seo, Chang-Hwan Im, Chang-Geun Heo, Jae-Kwang Kim, Hyun-Kyo Jung, Cheol-Gyun Lee.
"Multimodal Function Optimization Based on Particle Swarm Optimization", *IEEE Transactions on Magnetics, vol. 42, No. 4 (2006).*
School of Electrical Engineering and Computer Science, Seoul National University, Seoul.

[10] Jang-Ho Seo, Chang-Hwan Im, Chang-Geun Heo, Jae-Kwang Kim, Hyun-Kyo Jung, Cheol-Gyun Lee.
"An Improved Particle Swarm Optimization Algorithm Mimicking Territorial Dispute Between Groups for Multimodal Function Optimization Problems", *IEEE Transactions on Magnetics, vol. 44, No. 6 (2008).*
School of Electrical Engineering and Computer Science, Seoul National University, Seoul.

[11] S¨onke Hartmann, Dirk Briskorn.
"A Survey of Variants and Extensions of the Resource-Constrained Project Scheduling Problem", *The Open Access Publication Server of the ZBW – Leibniz Information Centre for Economics (2008).*
HSBA Hamburg School of Business Administration, Hamburg, Germany.

[12] Shahram Shadrokh, Fereydoon Kianfar.
"A genetic algorithm for resource investment project scheduling problem, tardiness permitted with penalty", *European Journal of Operational Research,Vol. 181, No. 1 (2007).*
Industrial Engineering Department, Sharif University of Technology, Tehran, Iran.

[13] Mohammad Khalilzadeh, FereydoonKianfar, Mohammad Ranjbar.
"A Scatter Search Algorithm for the RCPSP with Discounted Weighted Earliness-Tardiness Costs", *Life Science Journal (2011).*
Industrial Engineering Department, Sharif University of Technology, Tehran, Iran.

[14] Bert DE REYCK, Willy HERROELEN.
"The Multi-mode Resource Constrained Project Scheduling Problem With Generelized Precedence Relations", *European Journal of Operational Research, Vol. 119, No. 2 (2006).*
Katholieke Universiteit Leuven, Leuven, Belgium.

[15] Jan Böttcher, Andreas Drexl, Rainer Kolisch and Frank Salewski.
"Project Scheduling Under Partially Renewable Resource Constraints", *Management science, Vol. 45, No. 4 (1999).*
Institut für Betriebswirtschaftslehre, Christian-Albrechts-Universität zu Kiel, Olshausenstr, Germany.

[16] Ran Etgar, Avraham Shtub, Larry J. LeBlanc.
"Scheduling projects to maximize net present value — the case of time-dependent, contingent cash flows", *Eur. Journal of Operational research, Vol. 96, No. 1 (1997).*
Department of Industrial Engineering, Tel Aviv University, Ramat Aviv, Israel.

[17] PHILIPPE BAPTISTE, CLAUDE LE PAPE, WIM NUIJTEN.
"Satisfiability Tests and Time-Bound Adjustments for Cumulative Scheduling Problems", *Annals of Operations Research, Vol. 92, No.0 (1999).*
Université de Technologie de Compiègne.

[18] Jean Damay, Alain Quilliot, Eric Sanlaville.
"Linear Programming based algorithms for preemptive and non-preemptive RCPSP", *Eur. Journal of Operations Research. 182 (2007).*
Universit´e de Clermont II, Laboratoire LIMOS-CNRS UMR.

[19] James Kennedy, Russell Eberhart.
"Particle Swarm Optimization", *Proceedings of the IEEE International Conference on Neural networks, November 1995.*
Purdue School of Engineering and Technology Indianapolis, IN.

[20] Peter J. Angeline.
"Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and Performance Differences", *Proceedings of the 7th International Conference of Computer Science, California, USA, March 1998.*
Natural Selection Inc. 509.Colgate Street,Vestal, NY.

[21] David B. Fogel.
"An Introduction to Simulated Evolutionary Optimization", *Transactions on Neural Networks, Vol. 5, No. 1 (1994).*
Natural Selection Inc. La Jolla, CA.

[22] Steven Nahmias. ***"Production and Operations Analysis"***, [528-542].
Hebrew edition by the Open University of Israel, 2003.

[23] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein.
"Introduction to Algorithms", [165-206].
Hebrew edition by the Open University of Israel, 2008.

[24] אריה נחמיאס. ***"תורת המימון – ניהול פיננסי של גופים עסקיים"***, [37-53].
מהדורה שנייה, האוניברסיטה הפתוחה, 2008.

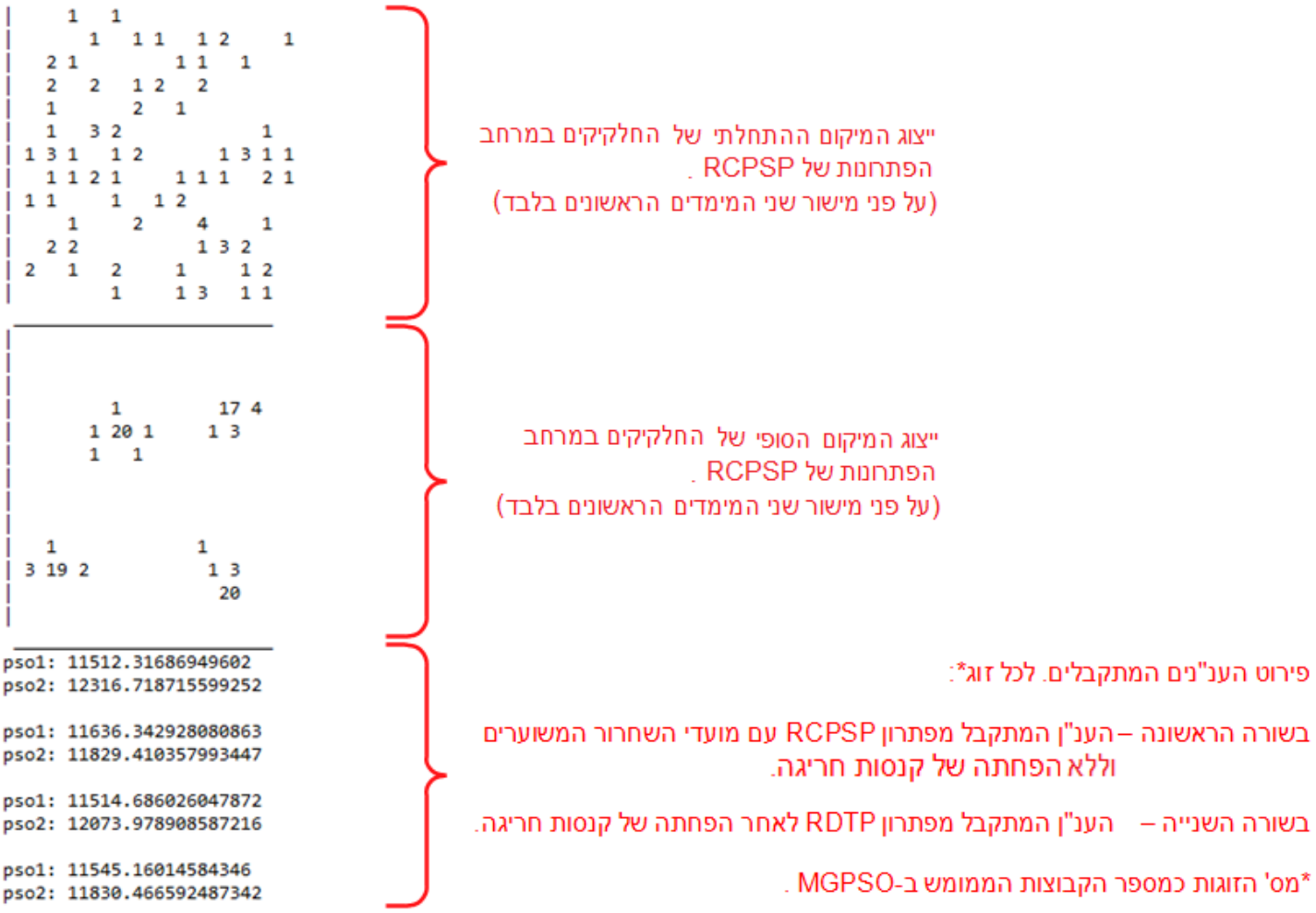
[25] PSPLIB - PROJECT SCHEDULING PROBLEM LIBRARY.
http://129.187.106.231/psplib/

11. נספחים

נספח א' – דוגמא לפלט של התוכנית שפותחה במסגרת הפרויקט

להלן דוגמא לפלט אמיתי אשר התקבל מריצה של התוכנית שפותחה בפרויקט זה, על רשת j901_1 .

הכיתוב והסימונים באדום מבטאים הערות ואינם חלק מהפלט עצמו.



Elapsed time: 13.340032881 sec — זמן הריצה

NPV of project: 12316.718715599252 — הענ"ן הסופי שהתקבל

Release Dates: RD-1: 18, RD-2: 48, RD-3: 73, RD-4: 98, — מועדי השחרור שנקבעו

Activity priority schedule: [3, 1, 18, 10, 44, 23, 28, 2, 4, 14, 6, 66, 11, 43, 29, 9, 5, 57, 7, 26, 22, 48, 67, 8, 24, • • •

תזמון הפעילויות שהתקבל
(הפעילות השמאלית ביותר משובצת ראשונה על הגאנט, הפעילות הבאה משובצת שנייה וכך הלאה)

זמני תחילת הפעילויות. בסוגריים המרובעות: זמן ההתחלה.

```
Start Times: 0:[0], 1:[0], 2:[0], 3:[0], 4:[1], 5:[23], 6:[10], 7:[25], 8:[33], 9:[3], ● ● ●
```

זמני סיום הפעילויות. בסוגריים המרובעות: זמן הסיום.

```
Finish Times: 0:[0], 1:[8], 2:[10], 3:[1], 4:[3], 5:[33], 6:[18], 7:[31], 8:[42], 9:[4], ● ● ●
```

```
Strain Graph for resource 1: (time of change in brackets)
|               XXX                   XX                XX                                  12
|                  XX                    XXX                                              11
|          XXXX     XXXXXXXXX   XXXXXXXXX     XXX                                         10
|                        XXX                        XX                                     8
|                                                XXXXX                                      7
|                                          XXXX                                             6
|                                                   X                                       5
|                                                                  XXXXX            XX      4
| X       XX                                                                               2
|                                                                                          1
 ____________________________________________________________________________________________
```

סקיצה של תרשים גאנט של רמות המשאב לאורך הפרויקט.

```
-->2(0) -->0(1) -->2(8) -->10(10) -->12(14) -->11(17) -->9(19) -->10(25) -->8(28) -->9(31) -->12(40) -->11(42) -->10(45) ● ● ●
```

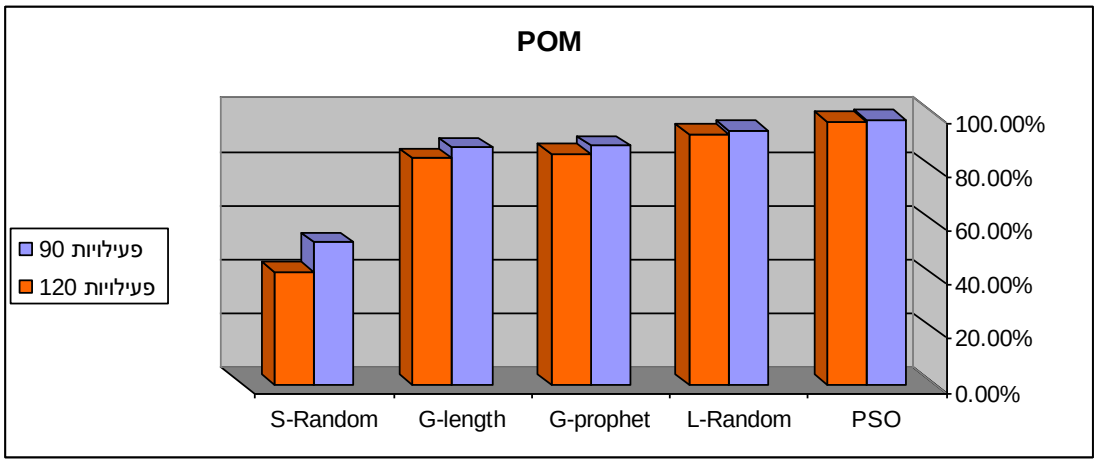
רמות המשאב. בסוגריים: זמן השינוי לרמה החדשה.

```
Strain Graph for resource 2: (time of change in brackets)
|                                  XXX    X                                                14
|                  XXXX           X                        XXX        XXXXX XXX            13
|                 X    XXX                    XXX                                         11
|                             XXXXXX   XXX                              X          XXXXXXXX  10
|                                      X        XXX        XXXXXXX          X              8
|         XXXXXXXX         XX                                                             7
|                                  XX           XXXX                        X              6
|                                               XXXXX                                      4
|                                                                                          3
|    X                                                                                     1
 ____________________________________________________________________________________________
-->0(0) -->2(3) -->0(4) -->7(8) -->0(16) -->11(17) -->13(18) -->11(22) -->7(25) -->0(27) -->10(28) -->12(34) -->6(35) -->10(37) ● ● ●
```

```
Strain Graph for resource 3: (time of change in brackets)
|                        XX                          X                                     17
|                    XX         XXXXXXXXXXXXXXX         XXXX XXXXX                       15
|  XXXXXXXXX              X                  XXXXX             XX         XXXXXX            14
|                                                 XXX                                      12
|                         X                                                                10
|                    XXXXX                                                                 9
| X                                                                                        7
|                                                                          XXXX XXXXXXXXXX  5
|                                                          XXXXXX  X      XX                3
|                                                                                          2
 ____________________________________________________________________________________________
-->7(0) -->13(1) -->0(10) -->15(16) -->9(18) -->14(23) -->17(24) -->10(26) -->16(27) -->15(33) -->16(35) -->13(42) ● ● ●
```

```
Strain Graph for resource 4: (time of change in brackets)
|                                  X                                                       13
|                                   XX   X                                 XXXXXXX        12
|  XX               XXXXXXXXX                      XXXXXX  XXX     X      XX              10
|                           XXXXX                                                          9
|                                     X                                                    8
|           XXXXXXXX         X             XXXXX      XX         XXXXXX                  7
|                                     X               XXXXX                                5
|                                  XXX                                          X  X       4
| X  XXXXX                                                              X                  3
|                                                                                          1
 ____________________________________________________________________________________________
-->3(0) -->10(1) -->3(3) -->0(8) -->6(10) -->7(14) -->11(18) -->10(23) -->7(27) -->9(28) -->13(33) -->12(34) -->4(36) ● ● ●
```

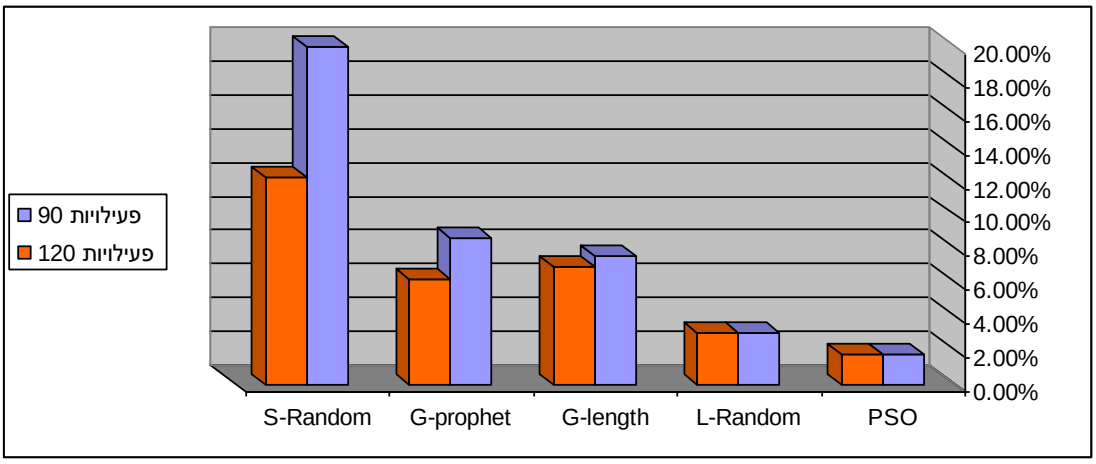
נספח ב' – גרפים הכוללים את תוצאות האלגוריתם האקראי הפשוט



גרף השוואת ממוצעי הפתרונות, לפי מדד POM .

סה"כ	120 פעילויות	90 פעילויות	
97.96%	97.70%	98.23%	**PSO**
93.98%	93.34%	94.61%	**L-Random**
87.46%	86.02%	88.90%	**G-prophet**
86.12%	84.16%	88.08%	**G-length**
47.39%	41.84%	52.95%	**S-Random**

ממוצעי הפתרונות, לפי מדד POM .



גרף השוואת סטיות התקן, לפי מדד POM .

סה"כ	120 פעילויות	90 פעילויות	
1.77%	1.78%	1.73%	**PSO**
3.08%	3.04%	3.01%	**L-Random**
7.47%	6.95%	7.57%	**G-length**
7.58%	6.18%	8.62%	**G-prophet**
17.44%	12.24%	19.94%	**S-Random**

סטיות התקן, לפי מדד POM.

נספח ג' – טבלת תוצאות המקסימום שהושגו על פני כל ההרצות במחקר

מקרא:

PSO – אלגוריתם ה-PSO .

G-proph – האלגוריתם החמדן לפי ערך הפעילות (Greedy by Prophet).

G-length – האלגוריתם החמדן לפי משך הפעילות (Greedy by Length).

L-Random – האלגוריתם האקראי הרחב (Large Random).

S-Random – האלגוריתם האקראי הפשוט (Simple Random).

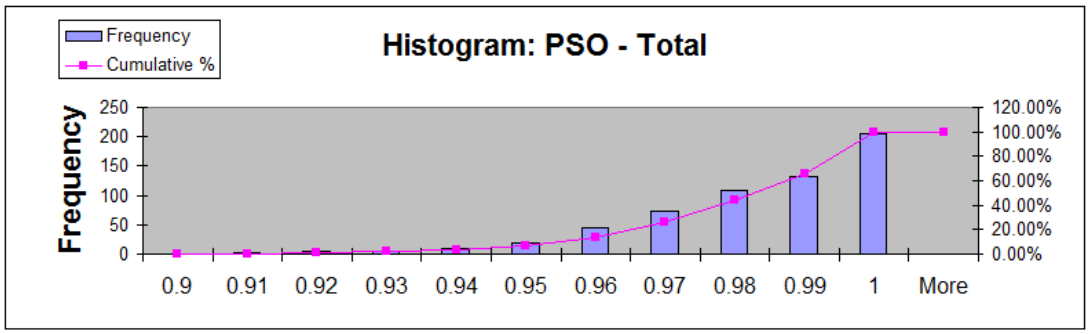
הצבע הצהוב מסמן את הערך שנבחר כמקסימום לחישוב ציון ה-POM .

MAX NPV	S-Random	L-Random	G-length	G-proph	PSO	רשת
11,185.78	5,854.43	10,497.49	9,528.47	8,095.08	11,185.78	**j1201_1**
17,199.59	10,953.32	16,627.17	14,143.36	14,439.87	17,199.59	**j1201_10**
16,413.33	9,086.10	15,041.99	12,356.07	14,330.93	16,413.33	**j1201_2**
8,337.22	1,139.96	8,165.30	6,145.86	6,200.75	8,337.22	**j1201_3**
7,302.66	1,261.16	6,895.77	5,342.76	5,888.38	7,302.66	**j1201_4**
18,518.30	11,278.63	17,921.37	16,730.42	17,079.69	18,518.30	**j1201_5**
9,897.03	3,054.26	9,673.03	8,059.93	9,621.98	9,897.03	**j1201_6**
9,663.03	3,318.58	9,051.27	6,399.00	7,270.08	9,663.03	**j1201_7**
10,625.93	4,854.82	10,625.93	8,470.46	8,650.17	10,587.78	**j1201_8**
9,996.73	4,520.06	9,479.54	7,356.87	7,924.17	9,996.73	**j1201_9**
13,783.95	5,722.87	13,336.56	11,078.13	12,005.91	13,783.95	**j1202_1**
13,594.63	7,227.82	13,136.13	11,691.34	11,455.20	13,594.63	**j1202_10**
12,499.75	4,891.61	11,964.53	10,416.26	10,201.63	12,499.75	**j1202_2**
14,345.88	9,065.49	13,846.28	12,830.08	12,359.89	14,345.88	**j1202_3**
18,477.23	10,060.06	17,874.95	16,246.13	15,239.45	18,477.23	**j1202_4**
6,189.34	1,690.45	5,677.46	5,326.42	5,581.12	6,189.34	**j1202_5**
14,766.79	8,348.33	14,531.38	13,147.20	12,741.92	14,766.79	**j1202_6**
15,135.27	6,938.84	14,690.27	12,408.16	12,514.85	15,135.27	**j1202_7**
12,207.47	4,354.42	11,902.94	11,277.58	10,115.61	12,207.47	**j1202_8**
7,108.33	1,514.93	7,028.27	6,147.77	6,261.04	7,108.33	**j1202_9**
19,119.64	8,579.32	18,503.73	15,608.30	17,671.18	19,119.64	**j1203_1**
9,462.93	3,373.97	8,873.40	8,338.54	9,064.80	9,462.93	**j1203_10**
17,267.68	5,971.55	16,540.67	16,151.29	14,823.50	17,267.68	**j1203_2**
13,821.88	6,899.61	13,608.82	13,301.52	12,424.80	13,821.88	**j1203_3**
21,022.73	9,342.18	20,581.33	19,690.14	18,450.24	21,022.73	**j1203_4**
20,940.29	10,848.20	20,465.12	17,820.16	19,060.72	20,940.29	**j1203_5**
9,514.04	2,540.80	9,416.70	7,535.52	8,151.48	9,514.04	**j1203_6**
14,923.35	6,481.19	14,863.08	13,115.80	13,692.80	14,923.35	**j1203_7**
21,171.46	11,240.48	20,712.13	19,084.36	18,684.02	21,171.46	**j1203_8**
9,145.90	1,640.58	8,968.13	7,805.20	8,820.36	9,145.90	**j1203_9**
11,957.24	7,041.95	11,639.33	10,199.94	7,261.13	11,957.24	**j901_1**
6,532.64	1,612.60	6,186.43	5,390.82	4,755.10	6,532.64	**j901_10**
16,262.93	10,652.77	15,726.47	13,149.36	15,053.26	16,262.93	**j901_2**
8,526.74	4,059.23	7,990.94	6,952.08	8,280.37	8,526.74	**j901_3**
9,823.45	6,650.57	9,609.57	6,226.16	7,698.74	9,823.45	**j901_4**
20,707.00	14,386.21	19,840.03	18,994.54	17,644.48	20,707.00	**j901_5**
11,583.86	7,281.54	11,210.07	9,519.38	10,222.73	11,583.86	**j901_6**

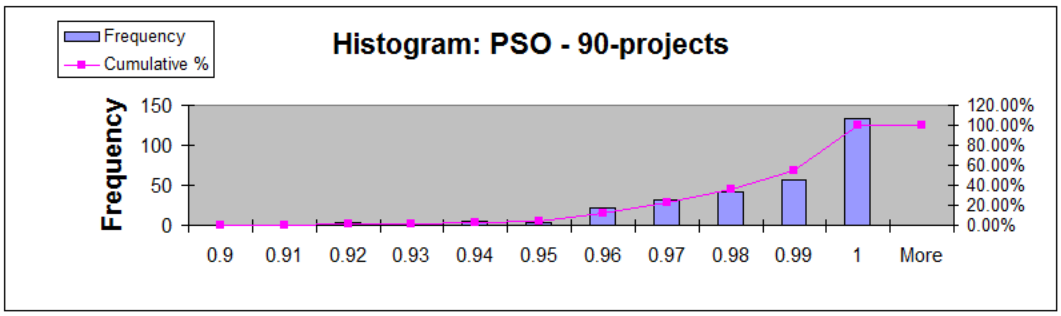
MAX NPV	S-Random	L-Random	G-length	G-proph	PSO	רשת
15,354.04	9,586.08	15,114.19	11,936.72	12,891.50	15,354.04	**j901_7**
11,367.26	7,785.44	10,782.13	9,384.77	8,685.43	11,367.26	**j901_8**
15,537.64	10,011.86	15,333.60	13,597.17	13,678.72	15,537.64	**j901_9**
7,991.04	3,875.85	7,749.90	7,458.60	7,265.94	7,991.04	**j902_1**
14,592.62	9,361.41	14,388.81	13,359.87	12,542.61	14,592.62	**j902_10**
8,442.53	4,773.86	8,195.99	7,901.43	8,442.53	8,359.58	**j902_2**
8,288.18	3,851.34	8,176.62	7,683.21	7,997.17	8,288.18	**j902_3**
18,339.02	12,287.44	18,159.26	16,537.18	17,343.95	18,339.02	**j902_4**
17,551.61	12,041.80	17,202.36	16,506.93	15,929.20	17,551.61	**j902_5**
3,787.64	-322.68	3,577.93	3,306.72	3,348.78	3,787.64	**j902_6**
8,703.71	4,733.66	8,479.41	7,217.40	6,797.61	8,703.71	**j902_7**
11,186.65	6,589.37	11,147.00	9,655.89	10,332.41	11,186.65	**j902_8**
13,335.91	8,651.69	13,335.91	11,252.01	11,917.73	13,255.85	**j902_9**
7,241.02	2,550.08	7,055.93	6,845.29	6,797.91	7,241.02	**j903_1**
12,572.03	5,912.49	11,972.19	11,613.04	11,050.93	12,572.03	**j903_10**
5,656.97	1,347.54	5,626.50	5,143.81	5,157.06	5,656.97	**j903_2**
7,988.88	3,179.59	7,699.41	7,988.88	7,840.39	7,884.10	**j903_3**
12,152.73	8,373.99	11,972.83	9,996.34	11,664.26	12,152.73	**j903_4**
11,603.13	7,387.18	11,174.29	11,476.66	10,708.60	11,603.13	**j903_5**
9,619.65	3,720.70	9,458.88	8,212.52	9,575.81	9,619.65	**j903_6**
17,680.04	12,531.97	16,903.63	16,119.91	15,680.94	17,680.04	**j903_7**
15,065.10	9,558.65	14,741.34	14,810.23	14,342.91	15,065.10	**j903_8**
15,066.43	6,945.11	14,766.39	14,578.86	14,143.33	15,066.43	**j903_9**

נספח ד' – היסטוגרמות ציוני ה-POM בחלוקה לאלגוריתמים וגודל פרויקט

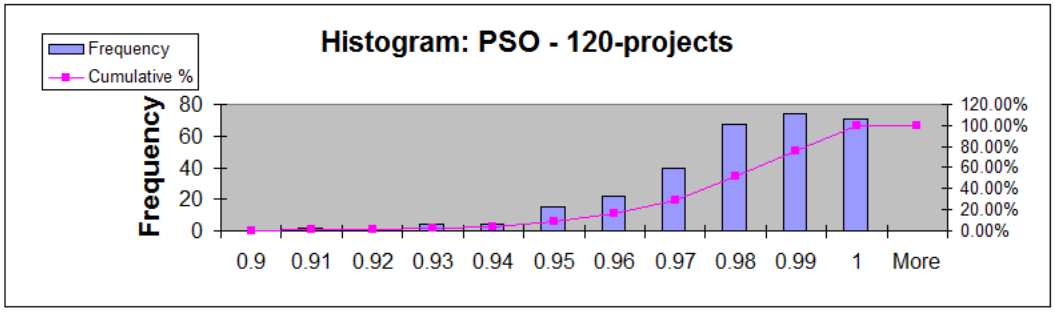
אלגוריתם ה-PSO



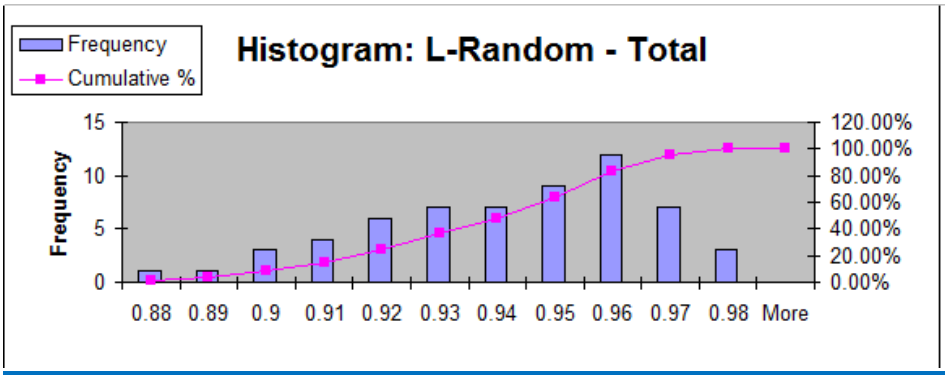
PSO - Total		
Cumulative %	*Frequency*	*Bin*
0.00%	0	0.9
0.33%	2	0.91
1.00%	4	0.92
1.83%	5	0.93
3.33%	9	0.94
6.50%	19	0.95
13.83%	44	0.96
25.83%	72	0.97
44.00%	109	0.98
65.83%	131	0.99
100.00%	205	1
100.00%	0	More



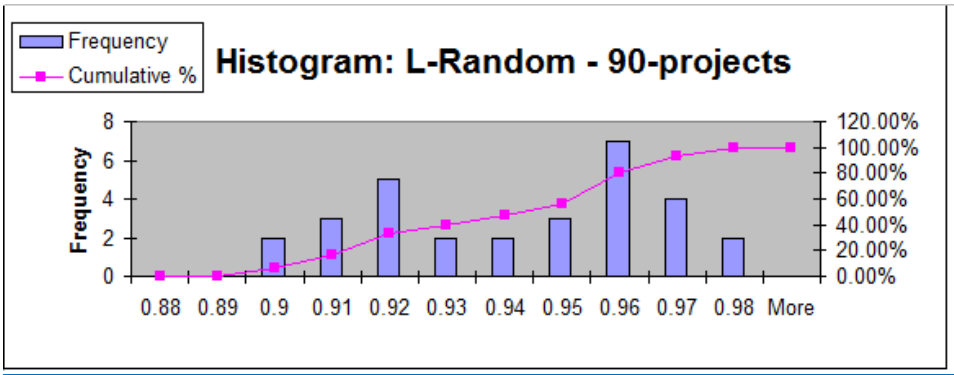
Cumulative %	*Frequency*	*Bin*
0.00%	0	0.9
0.00%	0	0.91
1.00%	3	0.92
1.33%	1	0.93
3.00%	5	0.94
4.33%	4	0.95
11.67%	22	0.96
22.33%	32	0.97
36.33%	42	0.98
55.33%	57	0.99
100.00%	134	1
100.00%	0	More



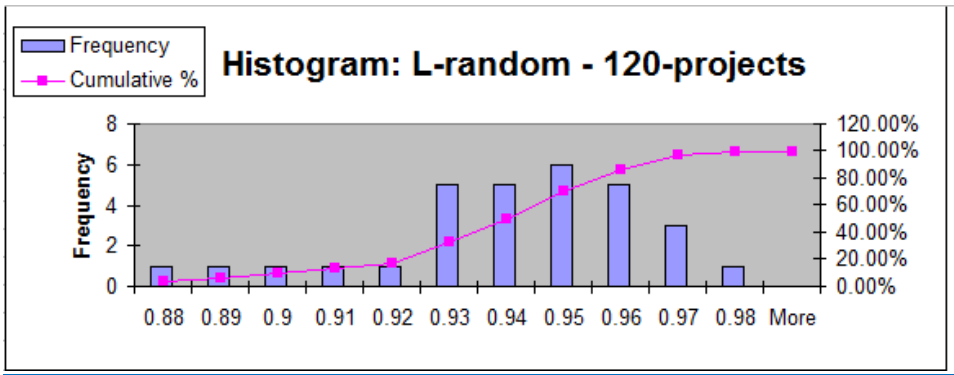
Cumulative %	*Frequency*	*Bin*
0.00%	0	0.9
0.67%	2	0.91
1.00%	1	0.92
2.33%	4	0.93
3.67%	4	0.94
8.67%	15	0.95
16.00%	22	0.96
29.33%	40	0.97
51.67%	67	0.98
76.33%	74	0.99
100.00%	71	1
100.00%	0	More



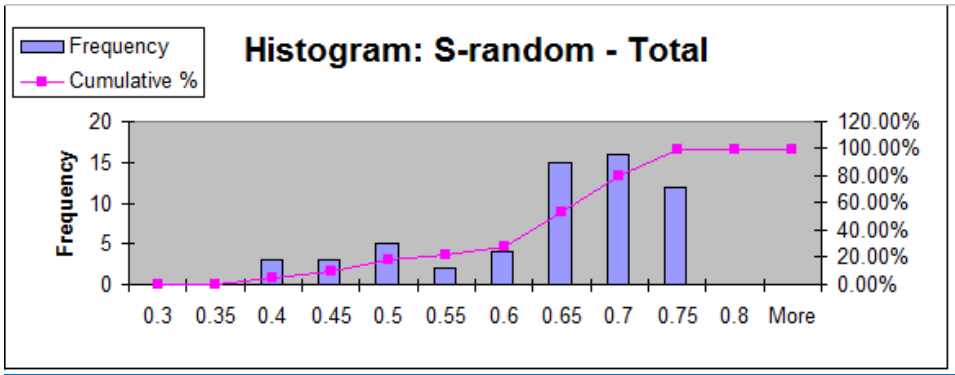
Cumulative %	Frequency	Bin
1.67%	1	0.88
3.33%	1	0.89
8.33%	3	0.9
15.00%	4	0.91
25.00%	6	0.92
36.67%	7	0.93
48.33%	7	0.94
63.33%	9	0.95
83.33%	12	0.96
95.00%	7	0.97
100.00%	3	0.98
100.00%	0	More



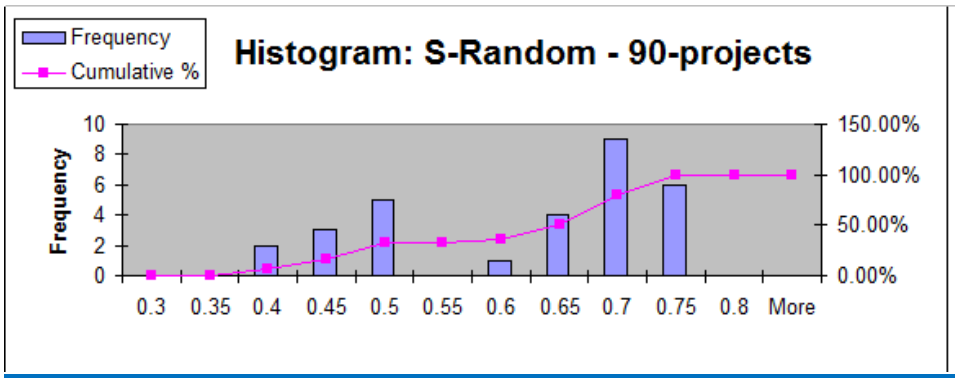
Cumulative %	Frequency	Bin
0.00%	0	0.88
0.00%	0	0.89
6.67%	2	0.9
16.67%	3	0.91
33.33%	5	0.92
40.00%	2	0.93
46.67%	2	0.94
56.67%	3	0.95
80.00%	7	0.96
93.33%	4	0.97
100.00%	2	0.98
100.00%	0	More



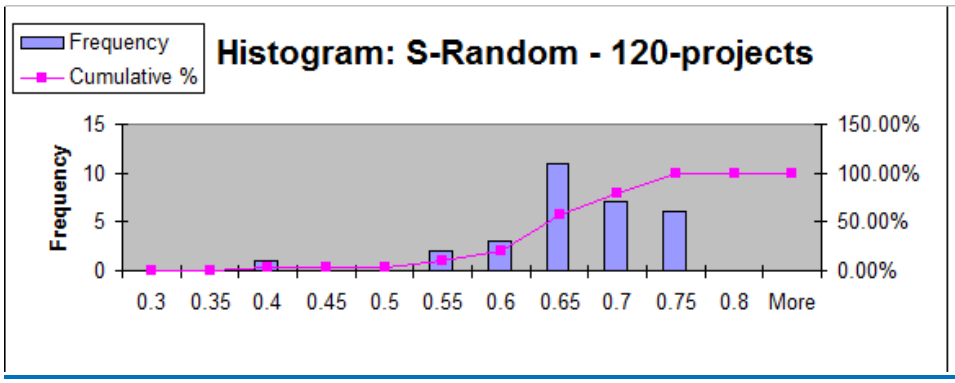
Cumulative %	Frequency	Bin
3.33%	1	0.88
6.67%	1	0.89
10.00%	1	0.9
13.33%	1	0.91
16.67%	1	0.92
33.33%	5	0.93
50.00%	5	0.94
70.00%	6	0.95
86.67%	5	0.96
96.67%	3	0.97
100.00%	1	0.98
100.00%	0	More



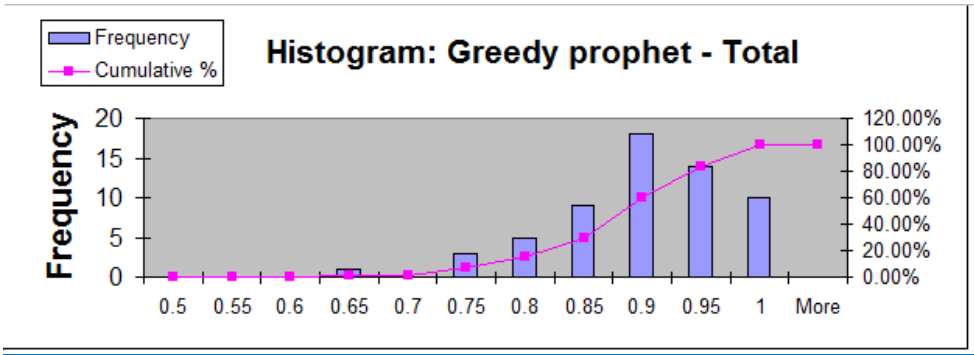
Cumulative %	Frequency	Bin
0.00%	0	0.3
0.00%	0	0.35
5.00%	3	0.4
10.00%	3	0.45
18.33%	5	0.5
21.67%	2	0.55
28.33%	4	0.6
53.33%	15	0.65
80.00%	16	0.7
100.00%	12	0.75
100.00%	0	0.8
100.00%	0	More



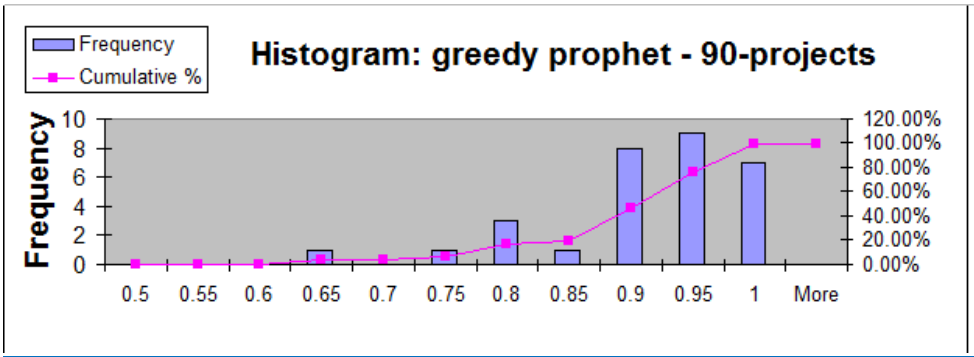
Cumulative %	Frequency	Bin
0.00%	0	0.3
0.00%	0	0.35
6.67%	2	0.4
16.67%	3	0.45
33.33%	5	0.5
33.33%	0	0.55
36.67%	1	0.6
50.00%	4	0.65
80.00%	9	0.7
100.00%	6	0.75
100.00%	0	0.8
100.00%	0	More



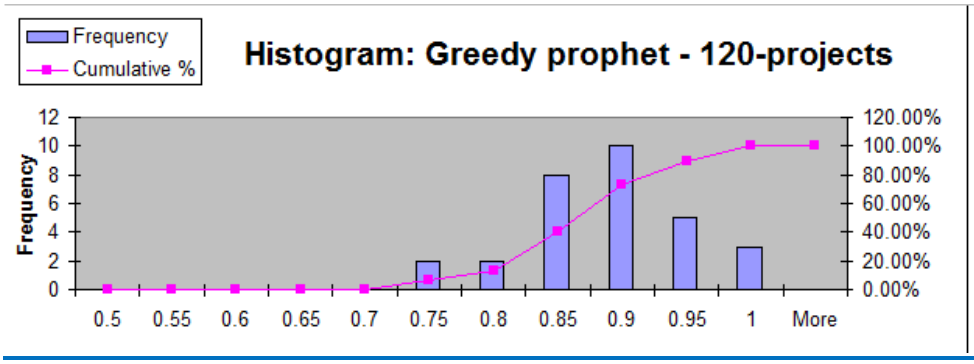
Cumulative %	Frequency	Bin
0.00%	0	0.3
0.00%	0	0.35
3.33%	1	0.4
3.33%	0	0.45
3.33%	0	0.5
10.00%	2	0.55
20.00%	3	0.6
56.67%	11	0.65
80.00%	7	0.7
100.00%	6	0.75
100.00%	0	0.8
100.00%	0	More



Cumulative %	Frequency	Bin
0.00%	0	0.5
0.00%	0	0.55
0.00%	0	0.6
1.67%	1	0.65
1.67%	0	0.7
6.67%	3	0.75
15.00%	5	0.8
30.00%	9	0.85
60.00%	18	0.9
83.33%	14	0.95
100.00%	10	1
100.00%	0	More

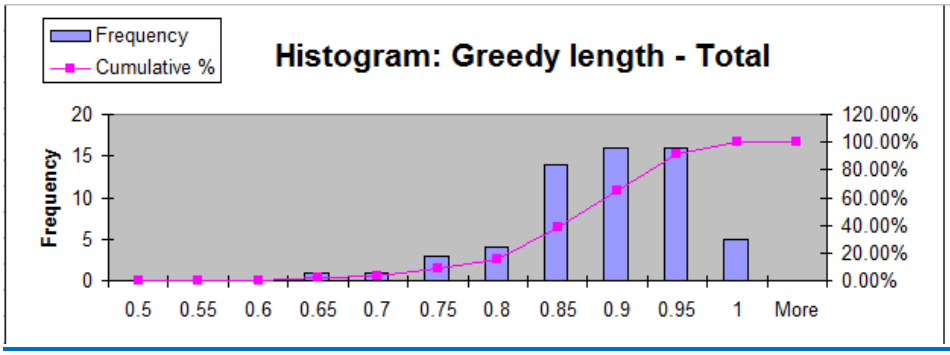


Cumulative %	Frequency	Bin
0.00%	0	0.5
0.00%	0	0.55
0.00%	0	0.6
3.33%	1	0.65
3.33%	0	0.7
6.67%	1	0.75
16.67%	3	0.8
20.00%	1	0.85
46.67%	8	0.9
76.67%	9	0.95
100.00%	7	1
100.00%	0	More

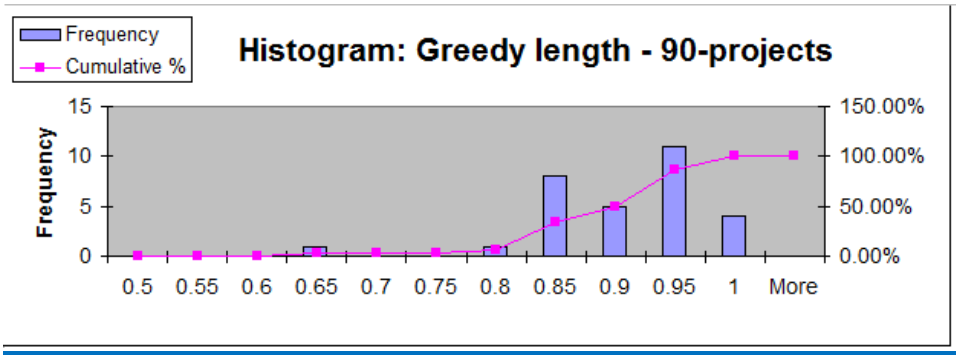


Cumulative %	Frequency	Bin
0.00%	0	0.5
0.00%	0	0.55
0.00%	0	0.6
0.00%	0	0.65
0.00%	0	0.7
6.67%	2	0.75
13.33%	2	0.8
40.00%	8	0.85
73.33%	10	0.9
90.00%	5	0.95
100.00%	3	1
100.00%	0	More

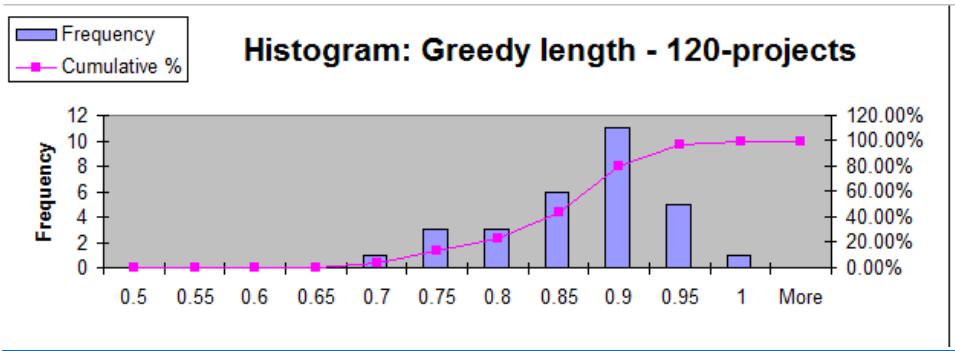
אלגוריתם חמדן לפי משך פעילות



Cumulative %	Frequency	Bin
0.00%	0	0.5
0.00%	0	0.55
0.00%	0	0.6
1.67%	1	0.65
3.33%	1	0.7
8.33%	3	0.75
15.00%	4	0.8
38.33%	14	0.85
65.00%	16	0.9
91.67%	16	0.95
100.00%	5	1
100.00%	0	More



Cumulative %	Frequency	Bin
0.00%	0	0.5
0.00%	0	0.55
0.00%	0	0.6
3.33%	1	0.65
3.33%	0	0.7
3.33%	0	0.75
6.67%	1	0.8
33.33%	8	0.85
50.00%	5	0.9
86.67%	11	0.95
100.00%	4	1
100.00%	0	More



Cumulative %	Frequency	Bin
0.00%	0	0.5
0.00%	0	0.55
0.00%	0	0.6
0.00%	0	0.65
3.33%	1	0.7
13.33%	3	0.75
23.33%	3	0.8
43.33%	6	0.85
80.00%	11	0.9
96.67%	5	0.95
100.00%	1	1
100.00%	0	More

נספח ד' – פירוט תכולת הדיסק המצורף

לספר פרויקט זה מצורף דיסק המכיל את החומר הנוסף הבא:

1. קוד המקור למימוש האלגוריתם המוצע ואלגוריתמי הקרה.

התוכנית נכתבה בשפת Java והינה בנויה מ-11 תתי-תוכניות. גם אלגוריתמי הבקרה מומשו ב-Java ויחד הם מכילים כ-2000 שורות קוד!

2. תוצאות ריצות האלגוריתמים.
3. רשתות הפרויקטים כפי שנלקחו מספריית PSPLIB.
4. נתוני הקלט, כולל נתוני תזרים מזומנים, כפי שהוכנסו לתוכניות המחשב.